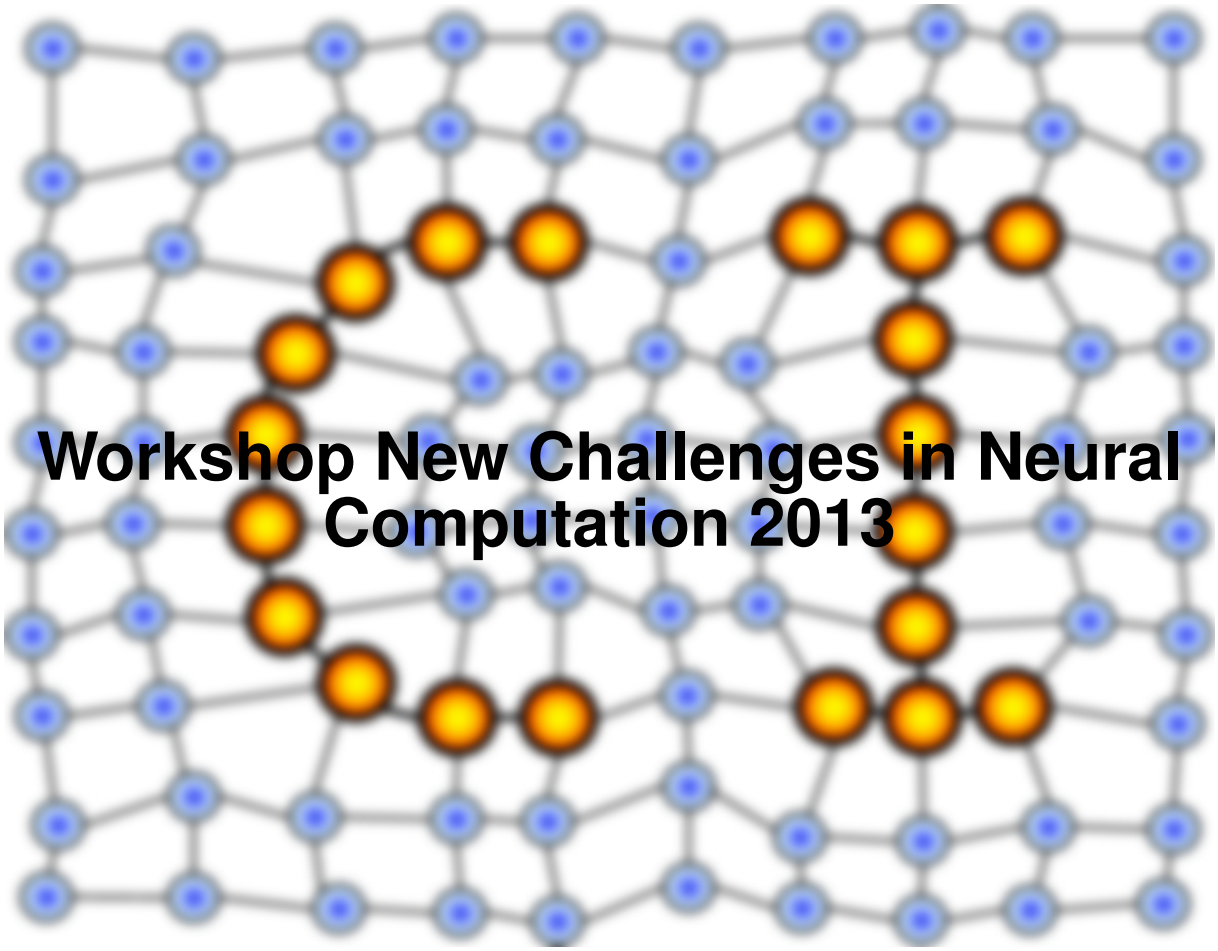
MACHINE LEARNING REPORTS



# Workshop New Challenges in Neural Computation 2013

Report 02/2013
Submitted: 21.08.2013
Published: 02.09.2013

Barbara Hammer[1], Thomas Martinetz[2], Thomas Villmann[3] (Eds.)
(1) CITEC - Centre of Excellence, University of Bielefeld, Germany
(2) Institute for Neuro- and Bioinformatics, University of Lübeck, Germany
(3) Faculty of Mathematics / Natural and Computer Sciences, University of Applied Sciences Mittweida, Germany

# Table of contents

# New Challenges in Neural Computation NC$^2$ – 2013

Barbara Hammer[1], Thomas Martinetz[2], and Thomas Villmann[3]

1 – Cognitive Interaction Technology – Center of Excellence,
Bielefeld University, Germany

2 – Institute for Neuro- and Bioinformatics, University of Lübeck, Germany

3 – Faculty of Mathematics / Natural and Computer Sciences,
University of Applied Sciences Mittweida, Germany

The workshop New Challenges in Neural Computation, NC$^2$, took place for the fourth time, accompanying the prestigious GCPR (former DAGM) conference in Saarbrücken, Germany. The workshop centers around exemplary challenges and novel developments of neural systems covering recent research concerning theoretical issues as well as practical applications of neural research. This year, fifteen contributions from international participants have been accepted as short or long contributions, respectively, covering diverse areas connected to data analysis, challenges in vision and robotics, prior knowledge integration, and local or sparse models, lots of the topics connecting to this years' focus topic on learning interpretable models with neural techniques. In addition, we welcome an internationally renowned researcher, Prof.Dr. Udo Seiffert from Fraunhofer IFF, Magdeburg, who gives a presentation about 'Challenges of high-dimensional data analysis from the application's perspective'. This invitation became possible due to the sponsoring of the European Neural Networks Society (ENNS) and the German Neural Network Society (GNNS). Following the workshop, a meeting of the GI Fachgruppe on Neural Networks and of the GNNS took place.

We would like to thank our international program committee for their work in reviewing the contributions in a short period of time, the organizers of GCPR for their excellent support, as well as all participants for their stimulating contributions to the workshop.

## Keynote talk: Challenges of high-dimensional data analysis from the application's perspective

**Udo Seiffert, Fraunhofer IFF Magdeburg**

**Abstract:**
The analysis of high-dimensional data typically leads to various challenges. This talk will address a number of these challenges against the background of neural computation / machine learning from the perspective of applications – primarily hyperspectral image processing. Apart from several general and versatile concepts a specific application might introduce further limitations in terms of the applicability of general concepts, but often also contributes additional a-priori information that eases prevailing problems. The demand for better interpretability of models and results becomes increasingly relevant.

# Application of Maximum Distance Minimization to Gene Expression Data

Jens Hocke, Thomas Martinetz

Institute for Neuro- and Bioinformatics, University of Lübeck

## 1 Introduction

The k-Nearest-Neighbor (k-NN) [1] algorithm is a popular non-linear classifier. It is simple and easy to interpret. However, the often used Euclidean distance is an arbitrary choice, because the data dimensions are not scaled according to their relevance. Similar to relevance learning in the context of LVQ classifiers [2], the scaling of the dimensions can be adapted by feature weighting to improve the classification rate of k-NN.

An optimal rescaling has to minimize the classification error $E(X)$ of the k-NN algorithm. Often this problem is called the *feature weighting* problem. We want to find a weight vector $\boldsymbol{w} \in \Re^D, w_\mu \geq 0, \mu = 1, ..., D$ for some given dataset $X = \{\boldsymbol{x}_i \in \Re^D, i = 1, ..., N\}$ that helps the classifier to minimize $E(X)$. In case the Euclidean distance is used, the weighted distance between two data points $\boldsymbol{x}, \boldsymbol{x}'$ becomes $d(\boldsymbol{x}, \boldsymbol{x}') = ||\boldsymbol{x} - \boldsymbol{x}'||_{\boldsymbol{w}} = \sqrt{\sum_{\mu=1}^{D} w_\mu (x_\mu - x'_\mu)^2}$.

Well known methods for feature weighting are Relief [3] and Simba [4]. Related is the more general problem of *metric learning* with Large Margin Nearest Neighbor Classification (LMNN) [5] as a popular approach, that optimizes the Mahalanobis distance $d(\boldsymbol{x}, \boldsymbol{x}') = ||\boldsymbol{x} - \boldsymbol{x}'||_W = \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^T W (\boldsymbol{x} - \boldsymbol{x}')}$.

We here present a method that contrary to the other methods is independent of the initial dimension scaling and evaluate it on gene expression data.

## 2 Maximum Distance Minimization

For rescaling the dimensions, we do not look at local neighbors, as the other methods do. Instead we try to minimize, by a very global optimization, the maximum distance between all pairs of data points of the same class, while keeping the pairwise distance between data points of different classes large. We therefore name our method Maximum Distance Minimization (MDM). Formally, we are solving the following constrained optimization problem

$$||\boldsymbol{x}_i - \boldsymbol{x}_l||^2_{\boldsymbol{w}} \geq 1 \; \forall i, l : y_i \neq y_l \tag{1}$$

$$||\boldsymbol{x}_i - \boldsymbol{x}_j||^2_{\boldsymbol{w}} \leq r \; \forall i, j : y_i = y_j \tag{2}$$

$$\min_{\boldsymbol{w}} r \qquad w_\mu \geq 0 \; \forall \mu, \tag{3}$$

where $y_i$, $y_l$, and $y_j$ are the class labels of $\boldsymbol{x}_i$, $\boldsymbol{x}_l$, and $\boldsymbol{x}_j$. The above problem can be formulated as a linear program, which is always solvable, even without slack variables.

| | Euclidean | MDM | Relief | Simba | LMNN |
|---|---|---|---|---|---|
| Breast | **8.07(6.13)** | 11.42(7.25) | 9.68(7.09) | 14.07(7.63) | 9.78(7.13) |
| Cancer | 1213.00(0.00) | 364.76(62.65) | 1213.00(0.00) | 1213.00(0.00) | 1136.96(0.75) |
| DLBCL | 13.11(5.24) | 14.67(5.33) | **11.17(5.03)** | 13.56(6.06) | 15.44(4.32) |
| | 661.00(0.00) | 293.86(34.13) | 661.00(0.00) | 661.00(0.00) | 559.54(1.99) |
| Leukemia | 2.21(2.27) | **1.74(1.96)** | 1.86(1.82) | 4.48(3.24) | 0.69(1.33) |
| | 985.00(0.00) | 473.24(55.28) | 985.00(0.00) | 984.94(0.24) | 822.50(4.77) |
| Lung | 4.37(2.77) | 5.49(3.18) | **4.22(2.66)** | 8.69(3.80) | 4.78(2.66) |
| Cancer | 1000.00(0.00) | 536.62(78.55) | 1000.00(0.00) | 999.78(0.42) | 870.86(1.87) |
| Novartis | 1.26(2.15) | **0.89(2.37)** | 0.98(1.98) | 3.81(4.43) | 0.39(1.34) |
| | 500.00(0.00) | 238.46(32.60) | 500.00(0.00) | 499.96(0.20) | 424.22(3.16) |

**Table 1.** Results for gene expression data. For comparisson we also included LMNN. The top entry is the average test error followed by the STD in parentheses. Below the error rates the average number of non-zero weights, again followed by the STD, is given.

## 3 Experiments

Experiments on UCI datasets show that MDM is independent of the initial scaling of the data dimensions [6]. Here we applied it to gene expression datasets available from the Broad Institute webside[1]. The data dimensions of each dataset were normalized so that the data points have zero mean and a variance of one. The k-NN (k=3) error rates in Table 1 were obtained by a 5-fold cross-validation that was repeated ten times. None of the tested methods is clearly better then any other and the variances are quite large. This shows how challanging this data is. There are only 70 to 250 samples and it has 500 to 1200 dimensions. Interestingly, MDM reduces the dimensionality heavily, which is worth to have a closer look at.

## References

1. Cover, T., Hart, P.: Nearest neighbor pattern classification. Information Theory, IEEE Transactions on **13**(1) (1967) 21–27
2. Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Netw. **15**(8-9) (2002) 1059–1068
3. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: Proc. 9th International Workshop on Machine Learning. (1992) 249–256
4. Gilad-Bachrach, R., Navot, A., Tishby, N.: Margin based feature selection - theory and algorithms. In: Proceedings of the twenty-first international conference on Machine learning. ICML '04, New York, NY, USA, ACM (2004) 43–50
5. Weinberger, K., Blitzer, J., Saul, L.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems 19, Cambridge, MA, MIT Press (2006)
6. Hocke, J., Martinetz, T.: Feature Weighting by Maximum Distance Minimization. In: ICANN 2013. (to appear)

[1] http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi

# Practical Estimation of Missing Phosphorus Values in Pyhäjärvi Lake Data

Alexander Grigorievskiy[1], Anton Akusok[1], Marjo Tarvainen[2], Anne-Mari Ventelä[2], and Amaury Lendasse[1,3,4]

[1] Aalto University, Department of Information and Computer Science, PO Box 15400, FI-00076 Aalto , Finland
{alexander.grigorevskiy,amaury.lendasse}@aalto.fi
[2] Pyhäjärvi Institute, Sepäntie 7, FIN-27500 Kauttua, Finland
{marjo.tarvainen,anne-mari.ventela}@pji.fi
[3] IKERBASQUE, Basque Foundation for Science, 48011 Bilbao, Spain
[4] Computational Intelligence Group, Computer Science Faculty, University of the Basque Country, Paseo Manuel Lardizabal 1, Donostia-San Sebastián, Spain

**Abstract.** Practical problem of missing values estimation of phosphorus concentration is addressed in this paper. There are several covariates which can be used to estimate phosphorus in Pyhäjärvi lake, however some of them also contain missing data. In addition, variable selection needs to be done in order to increase accuracy of modeling and facilitate understanding of underlying dependencies. We address the problem by first, Delta test variable selection and then by regression approach with Ridge Regression, SVM and LS-SVM accompanied with *wrapper* variable selection. It is shown that for some time periods it is possible to improve estimations from regression by averaging them with missing values imputation methods like Empirical Orthogonal Functions (EOF).

**Keywords:** Environmental Modeling, Missing values, Regression, Support Vector Machine, SVM, Least-Squares Support Vector Machine, LS-SVM, Empirical Orthogonal Functions, EOF

## 1 Introduction and Work Motivation

Pyhäjärvi lake is a large lake located on the south-west of Finland. The lake plays an important role in the local agriculture and fishing industries. Due to the human activity and changing climate the ecology of the lake has been challenged [1]. The main substance that influence the ecological balance in the lake is phosphorus. Therefore, it is very important to model and analyze phosphorus concentration in order to develop adequate measures for the lake protection.

Complication, which is frequently encountered when dealing with environmental data, is the presence of missing values. Measurements are often taken manually by humans and cases like spoiling the sample or sickness of a particular person are not exceptions. Selecting the best subset of covariates is also an important step in the modeling. In this work, the goal is to estimate concentration of phosphorus in various locations of Pyhäjärvi lake for the time period

26.03.1991 - 21.04.2008. Some values of phosphorus are given but many are missing. Some covariates also contain missing values. To shorten the exposition, data only for one location is considered in what follows.

The paper organization is the following: in the next section description of the dataset is provided. In the Section 3 regression approach to phosphorus concentration estimation is given. Then follows the missing values approach and finally conclusions.

## 2 Dataset Description

The dataset is shown in the Table 1. There are 16 variables (columns) and 1230 rows in the dataset. Each row correspond to averaged value of corresponding variable over 5 day interval. This interval is called "Week" for brevity. In the column "Complete dataset" number of present values of different variables is given. The variable to estimate is the second one - "Total P S11".

**Table 1.** Dataset and amounts of present values in variables

| | | Dataset | | |
|---|---|---|---|---|
| No. | Variable name | Complete dataset (1230 rows) | Part 1 (351 rows) | Part 2 (271 rows) |
| 1 | "Flow S11" | 1230 (full) | 351 (full) | 271 (full) |
| 2 | "Total P S11" | 227 | 59 | 58 |
| 3 | "Total P S10" | 225 | 60 | 58 |
| 4 | "Total P S12" | 226 | 58 | 72 |
| 5 | "Temperature" | 1228 | 351 (full) | 271 (full) |
| 6 | "Integrated Flow S11" | 1230 (full) | 351 (full) | 271 (full) |
| 7 | "Smoothed Flow S11" | 1230 (full) | 351 (full) | 271 (full) |
| 8 | "Rains" | 1230 (full) | 351 (full) | 271 (full) |
| 9 | "Sin Week" | 1230 (full) | 351 (full) | 271 (full) |
| 10 | "Cos Week" | 1230 (full) | 351 (full) | 271 (full) |
| 11 | "Time shift 1 Ph. S11" | 226 | 58 | 57 |
| 12 | "Time shift 2 Ph. S11" | 226 | 58 | 57 |
| 13 | "Time shift 1 Ph. S10" | 225 | 59 | 57 |
| 14 | "Time shift 2 Ph. S10" | 225 | 59 | 57 |
| 15 | "Time shift 1 Ph. S12" | 225 | 57 | 71 |
| 16 | "Time shift 2 Ph. S12" | 225 | 57 | 71 |

The sparsity of the dataset is 46% i. e. almost half of all the values are absent. However, missing values are distributed in time non uniformly. For variable "Total P S11" there are large periods (up to a year) when no data is present and periods where gaps are relatively small (several "Weeks"). Preliminary tests showed that missing values imputation methods provide good estimation of "Total P S11", when there are no large gaps between given values of this variable. Therefore, missing values imputation methods are applied only to datasets named "Part 1" (03.1991-02.1996) and "Part 2" (03.1997-12.2000) which correspond to time intervals with no big gaps between given values of "Total P S11". Regression modeling is conducted for the complete dataset.

Not all variables might be useful for phosphorus concentration estimation. One goal of this work is to select relevant variables and discard irrelevant.

**Regression Dataset.** Regression dataset (Table 2) is constructed from the complete dataset in the Table 1 by taking covariates where no missing data occurs (including "Temperature").

**Table 2.** Regression dataset

| No. | Variable name |
|---|---|
| **To predict** | **"Total P S11"** |
| 1 | "Flow S11" |
| 2 | "Temperature" |
| 3 | "Integrated Flow S11" |
| 4 | "Smoothed Flow S11" |
| 5 | "Rains" |
| 6 | "Sin Week" |
| 7 | "Cos Week" |
| 8 | "Rains Int. 1" |
| 9 | "Rains Int. 2" |
| ⋮ | ⋮ |
| 17 | "Rains Int. 10" |

The variable to predict is "Total P S11". The number of training samples is 227 and equals the number of present values in "Total P S11" variable. Having trained the regression model, it is possible to estimate phosphorus concentration on all other "Weeks" when it is missing. This is called regression approach and it is compared to missing values approach described in details in Section 4. Since missing values approach is studied only during periods "Part 1" and "Part 2", for all other "Weeks" regression approach is used to estimate "Total P S11". Ten additional variables No. 8-17 are added to the regression dataset. They are integrated values of "Rains" over 1 "Week" and so forth up to 10 "Weeks". The motivation for including these variables is to check possibility that phosphorus concentration depends on accumulated precipitation intensity during a long period.

In the following sections, we consider regression and then missing values approaches.

## 3 Regression Approach to Phosphorus Concentration Estimation

Regression approach has been applied to the data in Table 2. Three regression models are evaluated, and the best one which has smallest normalized mean square error (NMSE) is selected. First model is a linear one - Ridge Regression, and the other two are nonlinear Support Vector Regression (SVR) and Least-Squares Support Vector Regression (LS-SVR). Nonlinearity is obtained by using Gaussian kernel.

One thing that can deteriorate regression models is the presence of irrelevant, redundant, or too noisy input variables. Those can increase computational time, contribute to the curse of dimensionality and, finally, reduce accuracy of the regression [2]. In addition, selecting of only useful variables facilitates interpretability of the model.

### 3.1 Variable Selection

There exist many methods for variable selection. Overview of some of them is presented in [2] and [3]. These methods can be divided into three main categories: filters, wrappers and embedded methods. Filter methods optimize some external criteria and select a subset of input variables which corresponds to the optimum. Advantage of filter methods is that they are usually faster to compute than other types of methods, but disadvantage is that they doesn't take into account data model used during learning process. Wrapper methods utilize learning machine as a black box method to score different subsets of input variables. Multiple retraining of learning algorithm and measuring performance on

a separate validation set are usually required. This is a main disadvantage of this class of methods.

In this work, hierarchical variable selection is applied. On the first step less accurate but more computationally efficient filter method is used - Delta test [4],[5], on the second step, when less variables are left for analysis, wrapper method is utilized.

Based on the results of Delta test variables are divided onto 3 groups. First group is completely irrelevant variables which are discarded from subsequent investigation. The second group is important variables which are always kept. And finally to the third group attributed variables which are investigated through wrapper approach by passing all possible their combinations through the regression algorithm and measuring NMSE on validation set.

**Table 3.** Variable selection via Delta test for regression datasets

| No. of samples | Relevant variables | Variables to be investigated further |
|---|---|---|
| 227 | "Flow S11" , "Temperature", "Integrated Flow S11" | "Smoothed Flow S11", "Rains", "Sin Week", "Cos Week", "Rain int 1", "Rain int 4" |

Variables in the right most column are investigated further through a wrapper approach. Actually, three regression models are considered and selection of the best subset of variables is done along with selection of the best model.

### 3.2 Regression Models

Three regression models have been analyzed in this work. One linear - Ridge regression, and two nonlinear Support Vector Regression (SVR) and Least Squares Support Vector Regression (LS-SVR) [6].

Regularization parameter $\lambda$ in Ridge regression is adjusted via second internal cycle of cross validation. Gaussian kernel functions are used in SVR and LS-SVR. There are three hyper-parameters to adjust in SVR formulation: $C$ - regularization parameter, $\epsilon$ - width of a tube inside which no penalty for a point occurs, and $\sigma$ - width of a Gaussian kernel. We have utilized method of Cherkassky and Ma [7] followed by *pattern search* [8] to tune these parameters. LS-SVM Toolbox for Matlab has been used for LS-SVR modelling. Parameter optimization in this toolbox is done through coupled simulated annealing algorithm [9] and fine tuning through simplex method and cross-validation.

### 3.3 Regression Results

Before applying regression modeling all input variables and output variable have been normalized to have zero mean and unit variance. Generalization error of

different models and different subsets of input variables is measured by Monte-Carlo 15-fold cross-validation which is repeated 50 times. Number of folds is increased in comparison with standard 10 because number of samples in each dataset is small, and there is a need to increase number of samples for training. Regression model and subsets of variables which have the smallest NMSE are presented in the Table 4. We see that the best model is LS-SVM and the worst

**Table 4.** Relevant variables and best models for the regression dataset

| Best model | Relevant variables | $NMSE \pm (std)$ |
|---|---|---|
| LS-SVR | "Flow S11" , "Temperature", "Integrated Flow S11", "Smoothed flow S11", "Sin Week", "Cos Week" | $0.530 \pm (0.312)$ |
| Ridge R. | "Flow S11" , "Temperature", "Integrated Flow S11", "Sin Week", "Int. Rain 2", "Int. Rain 5" | $0.675 \pm (0.394)$ |
| SVM | "Flow S11" , "Temperature", "Integrated Flow S11", "Smoothed flow S11", "Sin Week", "Cos Week" | $0.570 \pm (0.359)$ |

one is Ridge Regression. This indicates the fact that dataset is highly nonlinear. SVM is the second best model. We suppose that the hyper-parameter selection strategy of LS-SVM toolbox is superior over the method we use for SVM.

The most relevant variables are the same for LS-SVM and SVM. Except "Rains" variable, all relevant variables form the application domain point of view are selected as important. Several subsets of variables are analyzed further in the missing values imputation approach.

## 4 Missing Values Approach to Phosphorus Concentration Estimation

Missing values datasets have been described in the Section 2. There are two datasets named "Part 1" and "Part 2". They correspond to time intervals when measurements of phosphorus are not very sparse. They include all 16 variables from the Table 1.

Regression modeling allows estimating phosphorus concentration when it is unknown. However, in regression modeling the sequential nature of the data is not taken into account. By utilizing missing values approach we are able to account for this and also include additional predictors (covariates) which themselves contain missing values. Importance of several subsets (Table 5) of input variables has been analyzed as well. Therefore, for periods for which missing values datasets are constructed, improved estimation of phosphorus concentration is obtained.

Generally, missing values imputation is a wide area of research with many applications [10], so it is hardly possible to try all the methods. Therefore, only

a subset from different classes of methods is selected and subsequent ensemble averaging is utilized to lighten possible disadvantages of a single method. Each method takes as input a matrix with missing values, fills missing values and returns the complete matrix. Due to the space constraints we describe only one method - Empirical Orthogonal Functions (EOF) [11]. It is a widely used method in meteorology and climate research for missing values imputation and is based on Singular Value Decomposition (SVD) (Algorithm 1). For other two: Mixture of Gaussians (MoF) [12],[13] and Singular Value Thresholding (SVT) [14] we redirect to the original articles.

**Algorithm 1** Empirical Orthogonal Functions

Given the incomplete matrix $\boldsymbol{X} \in \mathbb{R}^{m,n}$

1: Make initial imputation $\boldsymbol{X}^0$, for example, by column means
2: i = 0 (iteration number)
3: **repeat**
4: Perform SVD: $\boldsymbol{X}^i = \boldsymbol{U}^i \boldsymbol{D}^i (\boldsymbol{V}^i)^T$ to obtain $\boldsymbol{U}^i, \boldsymbol{D}^i$and$(\boldsymbol{V}^i)^T$
5: Nullify $K$ smallest singular values of $\boldsymbol{D}^i$. Denote this modified matrix as $\boldsymbol{D}_0^i$
6: Do inverse transformation: $\boldsymbol{X}_0^i = \boldsymbol{U}^i \boldsymbol{D}_0^i (\boldsymbol{V}^i)^T$
7: Restore exactly known values: $known(\boldsymbol{X}_0^i) = known(\boldsymbol{X}^0)$
8: i = i + 1 (iteration number)
9: **until** Convergence

### 4.1 Model Selection for Missing Values Approach

**Combining different models.** It is possible to select only one model based on the lowest NMSE of cross-validation, however there is s reason to keep all three and do an ensemble (*e.g.* see [15, p. 656]) Since regression can provide estimations of phosphorus it is also included in the ensemble.

Ensemble is done via arithmetic averaging of predictions from different models. However, even further improvement can be achieved if we choose which of the models to include in the averaging. There are five models we are investigating, namely "Regression", "Mixture of Gaussians 1 component" (MM1), "Mixture of Gaussians 2 components" (MM2), "SVT", "EOF".

**Experimental Setup.** Experiments are done in the similar way as regression experiments. Accuracy of imputation is characterized by Normalized Mean Squared Error (NMSE) and is measured by Monte-Carlo 15-fold cross-validation. There are 50 iterations in total, on each of those dataset is randomly permuted. The final estimation of NMSE is an average over folds within one iteration and total average over all iterations. Iterations of cross-validation are required because datasets are very small - only about 225 samples.

There are two missing values datasets "Part 1" and "Part 2" as described in Section 2. They are processed simultaneously in the cross-validation cycle. For each dataset, averaging estimations of all possible combinations of five models

and three subsets of variables (Table 5) is analyzed in terms of NMSE and standard deviation (STD) of NMSE.

### 4.2 Model Selection Results

Results of the model selection are presented in the Table 5. Three groups of variables which are interesting from the interpretation point of view have been analyzed. In particular, usefulness of "Rains" variable which has been rejected on the regression phase, as well as time shifted versions of phosphorus "Time shift 1 Ph. S11", "Time shift 2 Ph. S12".

**Table 5.** Groups of variables which have been tested for missing values imputation

| Missing Values Imputation Results | | | | |
|---|---|---|---|---|
| No. | Variable Name | Group 1 | Group 2 | Group 3 |
| 1 | "Flow S11" | ✓ | ✓ | ✓ |
| 2 | "Total P S11" | ✓ | ✓ | ✓ |
| 3 | "Total P S10" | ✓ | ✓ | ✓ |
| 4 | "Total P S12" | ✓ | ✓ | ✓ |
| 5 | "Temperature" | ✓ | ✓ | ✓ |
| 6 | "Integrated Flow S11" | ✓ | ✓ | ✓ |
| 7 | "Smoothed Flow S11" | ✓ | ✓ | ✓ |
| 8 | "Rains" | ✓ | | |
| 9 | "Sin Week" | ✓ | ✓ | ✓ |
| 10 | "Cos Week" | ✓ | ✓ | ✓ |
| 11 | "Time shift 1 Ph. S11" | ✓ | ✓ | |
| 12 | "Time shift 2 Ph. S11" | ✓ | ✓ | |
| 13 | "Time shift 1 Ph. S10" | ✓ | ✓ | |
| 14 | "Time shift 2 Ph. S10" | ✓ | ✓ | |
| 15 | "Time shift 1 Ph. S12" | ✓ | ✓ | |
| 16 | "Time shift 2 Ph. S12" | ✓ | ✓ | |
| Best model combination 10001: "Regression LS-SVM", "EOF" | | | | |
| $NMSE \pm std$, Part 1 | | $0.503 \pm 0.599$ | $0.504 \pm 0.634$ | $\mathbf{0.503 \pm 0.637}$ |
| $NMSE \pm std$, Part 2 | | $0.343 \pm 0.611$ | $0.340 \pm 0.665$ | $\mathbf{0.340 \pm 0.662}$ |

It turns out that the best model combination is an average of estimations of LS-SVM Regression and EOF. Actually, the best variable subset and best model combination is selected as compromise between two "Part 1" and "Part 2" datasets. The reason is that sometimes one model combination is better for "Part 1" and another one is better for "Part 2". So, resulting table is produced by manually inspecting NMSE and STD for various sets of variables and model combinations, and choosing the one with good results for both "Part 1" and "Part 2". Since the difference in NMSE and STD is not very large for the three groups, the third group might be preferred. This means that variable "Rains" and time shifted values of phosphorus are insignificant variables for this problem.

It can be observed that STD is higher than NMSE in all cases. This is an indicator of the fact that some extreme values of phosphorus concentration is very hard to predict.

## 5 Conclusions

Practical problem of phosphorus concentration estimation has been addressed in this article. Two stage approach has been developed where on the first stage regression problem with only complete covariates have been solved and on the second stage improvements by missing values method has been made. Selection of the best regression model and variable selection have been done along.

Empirical Orthogonal Functions(EOF) method in combination with LS-SVM achieved the best accuracy for predicting phosphorus concentration.

In the future, other classes of methods are intended to be applied for the problem. We plan to use existing methods or develop new ones which can do nonlinear regression with missing values in the covariates.

## References


1. Ventelä, A.M., Kirkkala, T., Lendasse, A., Tarvainen, M., Helminen, H., Sarvala, J.: Climate-related challenges in long-term management of säkylän pyhäjärvi (SW finland). Hydrobiologia **660** (2011) 49–58
2. François, D.: High-Dimensional Data Analysis. VDM Publishing (2008)
3. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3** (March 2003) 1157–1182
4. Guillén, A., Sovilj, D., Mateo, F., Rojas, I., Lendasse, A.: Minimizing the delta test for variable selection in regression problems. International Journal of High Performance Systems Architecture **1**(4) (2008) 269–281
5. Jones, A.: New tools in non-linear modelling and prediction. Computational Management Science **1**(2) (07 2004) 109–149
6. Suykens, J.: Least Squares Support Vector Machines. World Scientific (2002)
7. Cherkassky, V., Ma, Y.: Practical selection of svm parameters and noise estimation for svm regression. Neural Netw. **17**(1) (2004) 113–126
8. Marin-Galiano, M., Luebke, K., Christmann, A., Rüping, S.: Determination of hyper-parameters for kernel based classification and regression. Technical Report / Universität Dortmund, SFB 475 Komplexitätsreduktion in Multivariaten Datenstrukturen 2005,38 (2005)
9. de Souza, S.X., Suykens, J.A.K., Vandewalle, J., Bollé, D.: Coupled simulated annealing. IEEE Transactions on Systems, Man, and Cybernetics, Part A **40**(2) (2010) 320–335
10. Little, R.J., Rubin, D.B.: Statistical Analysis with Missing Data. 2 edn. Wiley (2002)
11. Preisendorfer, R., Mobley, C.: Principal component analysis in meteorology and oceanography. Developments in atmospheric science. Elsevier (1988)
12. Ghahramani, Z., Jordan, M.I.: Learning from incomplete data. Technical report, Cambridge, MA, USA (1994)
13. Eirola, E., Lendasse, A., Vandewalle, V., Biernacki, C.: Mixture of gaussians for distance estimation with missing data. In: Machine Learning Reports 03/2012. (2012) 37–45 Proceedings of the Workshop - New Challenges in Neural Computation 2012.
14. Cai, J.F., Candès, E.J., Shen, Z.: A singular value thresholding algorithm for matrix completion. SIAM J. on Optimization **20**(4) (March 2010) 1956–1982

15. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA (2006)

# Replacing the time dimension: A Self-Organizing Time Map over any variable

Peter Sarlin

Turku Centre for Computer Science – TUCS, Department of Information Technologies,
Åbo Akademi University, Joukahaisenkatu 3-5, 20520 Turku, Finland
Peter.Sarlin@abo.fi

**Abstract.** The Self-Organizing Time Map (SOTM) is a recently introduced adaptation of the Self-Organizing Map for visualizing dynamics in cluster structures, or visual dynamic clustering. This paper extends the use of the SOTM to visualize changes in cluster structures over any variable of ordinal, cardinal or higher level of measurement. The rationale and functioning of the SOTM over any variable is illustrated with two real-world cases related to cluster structures in welfare, poverty and development indicators for a global set of countries.

**Keywords:** Self-Organizing Time Map, cluster analysis, visual dynamic clustering

## 1 Introduction

The Self-Organizing Time Map (SOTM) [1] is a recently introduced adaptation of Kohonen's Self-Organizing Map (SOM) [2] for visual dynamic clustering. Clustering refers to the reduction of data into a smaller number of groups or mean profiles. Further, dynamic clustering refers to the same task, but with changes in the clusters over time. While there exist since the mid-20th century a plethora of methods for clustering, such as *k*-means, Ward's hierarchical and fuzzy *c*-means clustering, only recently was an approach denoted evolutionary clustering proposed for dynamic clustering [3]. Evolutionary clustering creates a sequence of clustering solutions with a balance between being faithful to current data and comparable with the previous clustering result. Chakrabarti *et al*. [3] relate the usefulness of such an approach to four tasks: (*i*) consistency (i.e., familiarity with the previous clustering), (*ii*) noise removal (i.e., increases in robustness due to a historical consistent clustering), (*iii*) smoothing (i.e., a smooth view of changes), and (*iv*) cluster correspondence (i.e., relation to historical context). The SOTM is a visual approach to evolutionary clustering by using dimensionality reduction for providing a low-dimensional representation of dynamic clusters. As is already hinted but not exploited in [3], an essential question remains: *Why should the SOTM be restricted only to illustrating differences over the time dimension?*

This paper extends the use of the SOTM to visualize changes in cluster structures over any variable of ordinal, cardinal or higher level of measurement. This can be exemplified by customer segmentation based upon demographic data. Instead of

exploring how customer segments change over time, we can explore how customer segments change over other variables, such as age, education level, purchasing amount., etc. The usefulness becomes self-evident when translating differences in cluster structures over any variable to the context of dynamic views with evolutionary clustering: (*i*) familiarity with the previous clustering, (*ii*) increases in robustness due to a context-consistent clustering, (*iii*) a smooth view of differences, and (*iv*) relation to context of the variable. The relevance of such an approach is highlighted by the fact that this is a common setting in a wide range of domains, where data are high-dimensional and large-volume, such as how risk indicators for financial entities change over their size or geographical location or how welfare and poverty indicators for countries change over some characteristic of the countries.

In the vein of the last example, this paper illustrates how cluster structures in welfare, poverty and development indicators for a global set of countries change over two variables: an index measuring fulfillment of the Millennium Development Goals (MDGs) and the share of population below the poverty line. After briefly presenting the SOTM and its counterpart over any variable in Section 2, we illustrate it by presenting two cases on a real-world dataset in Section 3. Section 4 concludes and illustrates applicability in other domains.

## 2 A SOTM over any variable

This section presents the standard SOTM, its counterpart over any variable and visualizations of the SOTM.

### 2.1 The SOTM

The SOTM uses the capabilities of the SOM for abstraction of patterns in data. In the form that the SOTM was presented in [1], it provides means for abstractions of temporal structural changes by illustrating how cross-sections evolve over time in one-dimensional SOMs [2]. Whereas time has been introduced to the SOM in numerous studies (e.g., Temporal SOM [4] and Merge SOM [5]), the objective of the SOTM is inherently different as it aims at visualizing how multivariate cross-sectional data evolve over time.

To observe the cross-sectional structures of the dataset for each time unit $t$ (where $t$=1,2,…,$T$), the SOTM performs a SOM-based mapping from the input space $\Omega(t)$, approximating the probability density functions $p(x,t)$ of time-restricted subsets of the data, onto a one-dimensional array $A(t)$ of output units $m_i(t)$ (where $i$=1,2,…,$M$). To preserve the orientation between consecutive patterns, the SOTM uses short-term memory by initializing reference vectors of $A(t\text{-}1)$ with those of $A(t)$. However, the first eigenvector of principal component analysis (PCA) on $\Omega(t_1)$ is used for an initialization of $A(t_1)$. Adjustment to temporal changes is achieved by performing a SOM-type batch update per time unit $t$. Thereafter, the timeline is created by arranging all $A(t)$ in an ascending order of time $t$. The topology preservation of the SOTM is hence twofold: the horizontal direction preserves time topology and the vertical preserves data topology. While being time restricted, the parametrization of the SOTM follows the two standard steps from the SOM paradigm. First, it locates

best-matching units (BMUs) by a time-restricted matching of each data to the unit with shortest Euclidean distance, i.e. $\min\left\|x(t) - m_i(t)\right\|$, and then it updates each reference vector $m_i(t)$ through a time-restricted version of the batch formula. The number of updates and size and type of neighborhood can obviously be adapted depending of the purpose of use and characteristics of the data at hand.

### 2.2 Replacing time in the SOTM

The above description of the SOTM focuses on visual dynamic clustering. This section draws upon the approach of the SOTM, yet replaces the *x*-dimension of time *t* with any variable *v*. For instance, similarly as time can be used for dividing data into 10 distinguishable time points, the variable *v* can be used to divide data into separate subsets. Hence, this section presents a SOTM in which the time dimension *t* has been interchanged to a variable *v*, which eventually implies a SOTM for illustrating changes in cluster structures over variables.

The properties of variable *v* (where $v$=1,2,…,$V$) follow those of time *t* in the original SOTM by being a discretized ordinal or cardinal variable (or higher level of measurement), having arbitrary frequency and being related to all entities in $\Omega$. In particular, this excludes the use of nominal variables with values which have no numerical meaning (e.g., gender and occupation), and does thus not enable ordering or ranking of data. Thus, the variable *v* is simply a transformation of dataset $\Omega$ into subsets $\Omega(v)$ according to the discrete values of variable *v*.

As replacing *t* with *v* involves no changes in the overall procedure, we mainly focus herein on the implications for the two SOM-based steps in SOTM training. Thus, for each $A(v)$, a data points $x_j(v) \in \Omega(v)$ (where $j$=1,2,…,$N(v)$) are compared to reference vectors $m_i(v) \in A(v)$ and assigned to their BMUs $m_c(v)$:

$$\left\|x(v) - m_c(v)\right\| = \min_i \left\|x(v) - m_i(v)\right\|, \tag{1}$$

Then, each reference vector $m_i(v)$ is adjusted using the batch update formula:

$$m_i(v) = \frac{\sum_{j=1}^{N(v)} h_{ic(j)}(v) x_j(v)}{\sum_{j=1}^{N(v)} h_{ic(j)}(v)}, \tag{2}$$

where index *j* indicates the input data that belong to unit *c* and the neighborhood function $h_{ic(j)}(v) \in (0,1]$ is defined as a Gaussian function

$$h_{ic(j)}(v) = \exp\left(-\frac{\left\|r_c(v) - r_i(v)\right\|^2}{2\sigma^2}\right), \tag{3}$$

where $\|r_c(v) - r_i(v)\|^2$ is the squared Euclidean distance between the coordinates of the reference vectors $m_c(v)$ and $m_i(v)$ on the one-dimensional array, and $\sigma$ is the user-specified neighborhood parameter.

In this vein, the functioning principles of the SOTM over variables $v$ can be summarized as follows:

```
v = 1
initialize A(v) using PCA on Ω(v)
apply the batch update to A(v) using Ω(v)
while v<V
    v = v + 1
    initialize A(v) using the reference vectors of A(v-1)
    apply the batch update to A(v) using Ω(v)
end
order A(v) in an ascending order of variable v
```

### 2.3 Visualizing any SOTM

Visualizing the SOTM, independent of whether it is computed over time or any other variable, can make use of the same set of visualization aids. We focus in this paper on two representations of the SOTM.

The multidimensionality of the SOTM can be represented using feature planes, as is common in the SOM and SOTM literature. They are views of the spread of values for each variable on the SOTM grid and enable assessing the variation in cross-sectional distributions over the chosen variable $v$. Herein, feature planes make use of ColorBrewer's scale [6], where variation of a blue hue occurs in luminance. Hence, light to dark represent low to high values, as is exemplified in Figs. 1 and 3.

For representing the multivariate structures of a SOTM, we can use a Sammon's mapping-based coloring. Sammon's mapping [7] is one of the seminal multidimensional scaling methods that stresses local pairwise distances. Over the chosen variable $v$, this approach enables exploring structural properties of $v$ (vertically) and differences in structures (horizontally) by trying to match the pairwise distances of the SOTM units with their distance in the high-dimensional space. This can be represented in two ways. First, a Sammon's mapping of the SOTM plots all units ($m_i(v)$ where $v$=1,2,…,$V$) to one dimension and assigns the value of the Sammon's dimension to each unit. Variable $v$ is disentangled plotting the Sammon's dimension individually for each element of variable $v$. Thus, this representation has Sammon's dimension on the $y$ axis and variable $v$ on the $x$ axis, in which adjacent units are connected with solid (data topology) and dashed (variable topology) lines for a net-like representation. Second, the Sammon's dimension may be used as an input to a coloring method by Kaski *et al.* [8] for visualizing cluster structures of the SOTM. The Sammon's dimension of the SOTM units is paired with the uniform color space CIELab [9], where perceptual differences of colors represent distances in data. However, as also the individual SOMs of the SOTM are one-dimensional, we only use one color dimension (blue to yellow) to represent differences between units. Hence, the coloring represents distance structures on the SOTM grid, where distances in color show differences among clusters and their differences over variable $v$.

## 3 Experiments

In this section, we present the SOTM over cardinal variables on a real-world dataset. Following the case in [1], we also illustrate an application of the SOTM on a multivariate dataset of development and welfare indicators with patterns for a large number of economies over the past two decades. The dataset includes a selection of World Bank's World Development Indicators for tracking the progress of the MDGs. The eight MDGs represent commitments to reduce, by 2015, hunger and poverty, and to tackle gender inequality, ill-health and lack of education and access to clean water, as well as environmental degradation. The dataset $\Omega$ is in panel format, where rows represent country-year observations, including 207 countries spanning from 1990–2008, and columns represent 15 indicators measuring fulfillment of the MDGs, which are all transformed using min-max normalization. However, instead of years, we show how cluster structures differ over two cardinal variables: an MDG index and population below the poverty line.

### 3.1 MDG indicators over an MDG index

The first application on the MDGs illustrates how cluster structures evolve over a broad MDG index. The MDG index is based upon work in [10], where a large set of welfare, poverty and development indicators were aggregated to an overall measure of MDG fulfillment. While a large number of measures are highly correlated, such as ill-health and poverty, some are less so, such as poverty and environmental degradation or gender inequality. As the values of the MDG index are centered around 0.4, although being in the interval of [0,1], we transform them into percentiles to better represent how the cluster structures evolve over the MDG index. Further, to discretize $v$, we focus on deciles, and thus $v$=0.1,0.2,…,1.0.

The architecture of the SOTM is chosen to be 6x10 units, where six units represent data topology for each of the ten deciles of the MDG index. Fig. 1 visualizes univariate changes in cluster structures over the MDG index. It illustrates some obvious patterns, such as decrease in poverty measures over the MDG index, whereas some less so. For instance, seats held by women in parliament take average values for MDG index values of [0.2,0.4] and high for [0.8,1.0]. Likewise, high CO2 emissions and low official development assistance (ODA) characterize MDG index values of [0.4,0.6]. This indicates that polluting countries with little gender equality and ODA, but good values in poverty and health-related measures, populate the mid part of the SOTM. Moreover, one can also observe that while the share of underweight children decreases abruptly over the MDG index, the HIV rate and the tuberculosis prevalence rate decrease gradually over the cluster structures.

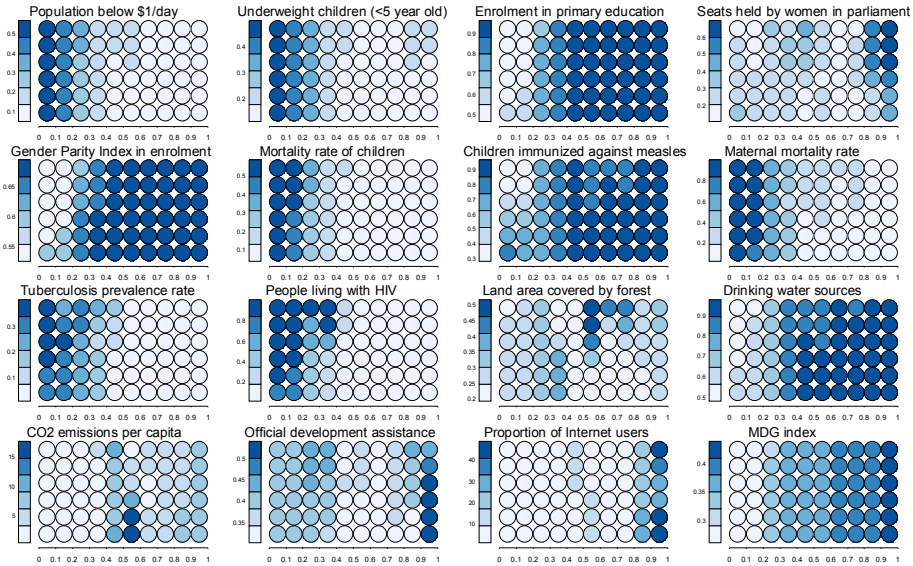



**Fig. 1.** Feature planes of the SOTM over the MDG index.

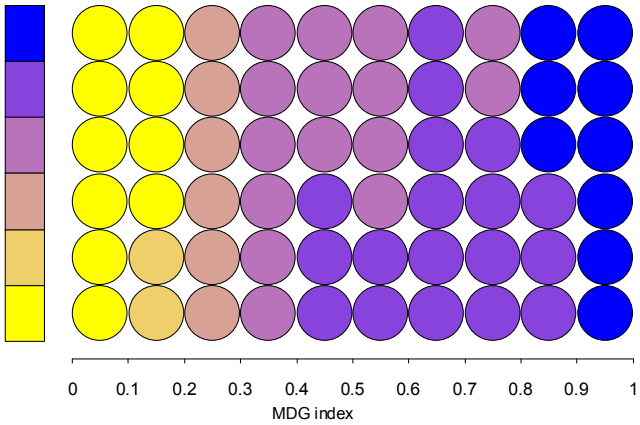



**Fig. 2.** Cluster coloring of the SOTM over the MDG index.

Fig. 2 shows, on the other hand, how the multivariate structures evolve over the MDG index. It shows that the largest differences occur in the lower end of the MDG index, whereas changes in structures for an MDG index of [0.3,1.0] are more gradual.

### 3.2 MDG indicators over population below the poverty line

In the second application, we use the same data, but interchange the MDG index to the share of the population below the poverty line. Again, we transform the variable $v$ into deciles, i.e., $v$=0.1,0.2,…,1.0.

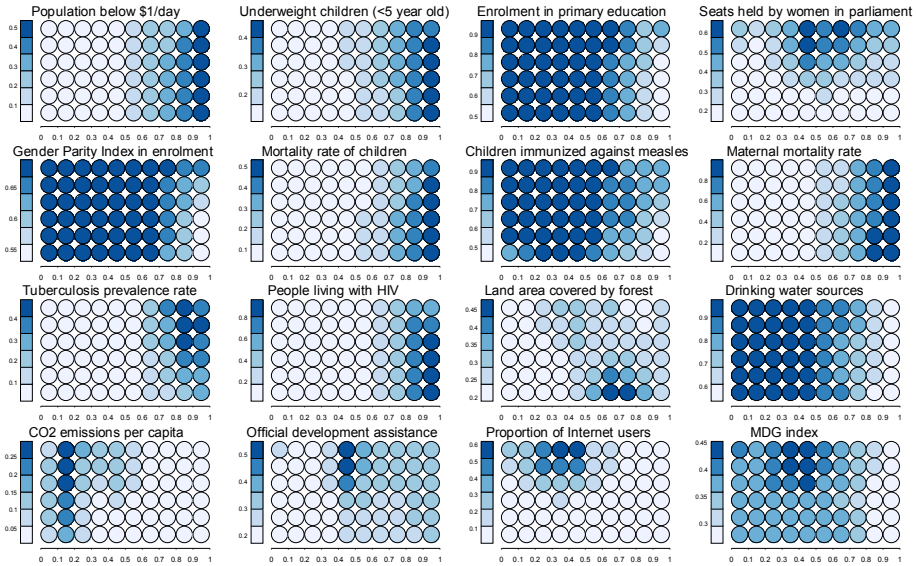



**Fig. 3.** Feature planes of the SOTM over the share of population below the poverty line

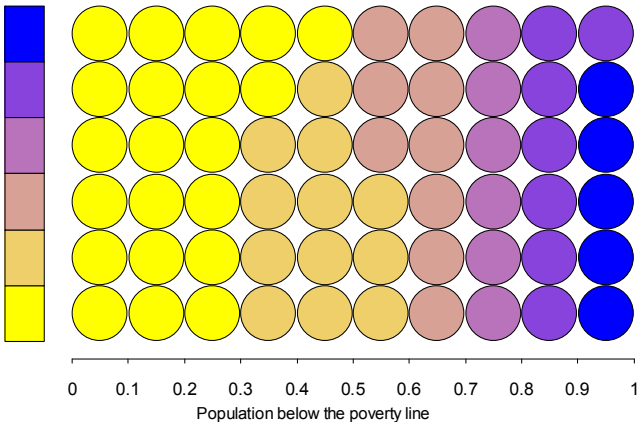



**Fig. 4.** Cluster coloring of the SOTM over the share of population below the poverty line

We follow the above application by again setting the architecture of the SOTM to 6x10 units, where six units represent data topology for each of the ten deciles of the population below the poverty line. As above, Fig. 3 visualizes univariate changes in cluster structures over the population below the poverty line. As the nature of variable *v* is different (i.e. not an index value), the changes in cluster structures are more gradual. Here, on the other hand, we can observe that countries with high CO2 emissions and little gender equality and ODA lie in the very beginning of the SOTM, whereas countries with higher proportion of internet users and more ODA and women in parliament are located in [0.3,0.5]. In Fig. 4, we can again observe that larger differences in cluster structures occur in less developed nations, which also

corresponds to the fact that population below the poverty line varies little below the median (as is seen in the first feature plane of Fig. 3).

## 4 Conclusions

This paper has illustrated how the SOTM can be used for an abstraction of changes in cluster structures over any variable. Whereas the standard SOTM performs visual dynamic clustering with a focus on changes over time, the present paper shows that the only restriction on the variable over which differences in cluster structures can be shown is ordinal, cardinal or higher level of measurement. The case examples used to illustrate the SOTM over any variable relate to welfare and poverty indicators, particularly to changes in cluster structures over an MDG index and the share of population below the poverty line. The simple demonstrations in this paper function as examples for the SOTM to be applied to other domains and problems. It is worth to note that it is not rare for cluster structures to change over various variables in most tasks, and hence this approach is expected to be useful also in a wide range of domains.

## References

[1] Sarlin, P.: Self-Organizing Time Map: An Abstraction of Temporal Multivariate Patterns. Neurocomputing, 99(1), 496–508 (2012)

[2] Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 66, 59–69 (1982)

[3] Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary clustering. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 554–560. ACM, Philadelphia, PA, USA (2006)

[4] Chappell G., Taylor J.: The temporal Kohonen map. Neural Networks 6, 441–445 (1993)

[5] Strickert M., Hammer B.: Merge SOM for temporal data, Neurocomputing, 64: 39–72 (2005)

[6] Harrower, M.A., Brewer, C.A.: ColorBrewer.org: An Online Tool for Selecting Color Schemes for Maps. The Cartographic Journal 40(1) 27–37 (2003)

[7] Sammon, J.W.: A nonlinear mapping for data structure analysis. IEEE Transactions on Computers 18, 401–409 (1969)

[8] Kaski *et al*. (2000) Kaski, S., Venna, J., Kohonen, T.: Coloring that reveals cluster structures in multivariate data. Australian Journal of Intelligent Information Processing Systems 6, 82–88 (2000)

[9] Colorimetry, 2nd Ed. CIE Publication No. 15.2 (1986)

[10] Sarlin, P.: Visual Tracking of the Millennium Development Goals with a Fuzzified Self-Organizing Neural Network. International Journal of Machine Learning and Cybernetics 3(3), 233--245 (2012)

# Image-Based Classification of Websites

Anton Akusok[1], Alexander Grigorievskiy[1], Amaury Lendasse[1,2,3], Yoan Miche[1]

[1] Department of Information and Computer Science,
Aalto University School of Science, FI-00076, Finland
[2] IKERBASQUE, Basque Foundation for Science, 48011 Bilbao, Spain
[3] Computational Intelligence Group,
Computer Science Faculty, University Of The Basque Country,
Paseo Manuel Lardizabal 1, Donostia/San Sebastian, Spain

**Abstract.** This paper presents a simple and powerful approach to image-based classification of webpages. It distinguishes between arbitrary classes, and is resilient to noise. The approach consists of extracting colour image descriptors, building a unified image representations, and running a fast classifier on top of it. The results are good even in the presence of noise, and excellent for some classes. Results on a subset of Caltech101 are given for comparison.

## 1 Introduction

Analysis of web content is an old task, emerged with the first Internet search engines. Being able to describe or classify a webpage is essential for various tasks like returning relevant search results [9], finding similar pages [13] or blocking unwanted or dangerous websites [14] like phishing ones.

Traditional webpage analysis approaches rely on textual information (text body of a page, address, keywords and links). But with the increase of bandwidth, storage and processing power, image data found heavy usage in webpages being a native to humans powerful expressive format. A modern user will probably be surprised seeing a text-only webpage without visual design.

Image data, while being an important source of information in the web, is hard for machine processing due to its extreme variability. The task can be simplified by restricting it to image classification in several classes of interest. Existing classification methods include target-specific ones [29], which cannot be generalized on arbitrary classification. An example of these are adult content detection methods based on the amount of skin colour in the image [20]. Other methods are very complicated and aimed on image understanding with object extraction and recognition [27, 6]. They are common in regular competitions like PASCAL VOC [7] or common benchmark datasets like Caltech101 [8]; otherwise they are impractical due to complicated modification and adaptation, a lack of ready-made toolboxes, and long training and running times.

The goal of the research is to create a universal method for image classification of arbitrary classes. Because exact classes of particular webpage images are unknown, a specific assumption is made that they have the same class as the

page itself, which may lead to a high amount of noise. Also the classifier should scale well with the size of a dataset, because huge datasets are easily obtained from the web, and are a key for overcoming the noise problem.

The rest of the paper is organized as following. The next section describes the four stages of the proposed methodology for image-based classification. The *Experiments and Results* section presents classification results for the original web image dataset, as well as for the similar subset of Caltech101 images for comparison. And the last section concludes on the performed work.

## 2 Methodology

The current goal of methodology for website image-based classification is to develop a uniform approach to general image-based classification. It allows estimating the feasibility of such a classification for each class of interest, adapting the system to changing requirements, and leaves space for later modifications. The proposed approach, satisfying the aforementioned restrictions and having clear and feasible to implement image processing pipeline is depicted on Figure 1.

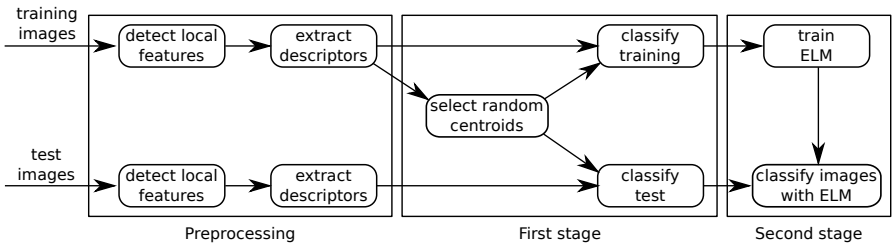



**Fig. 1.** Overview of image classification method. Three major stages are pre-processing, building image representations with random vector quantization, and final classification step with ELM. Image classification results are combined into website classes as described in the last section of the methodology.

The pipeline consists of three image classification stages, plus a website combination step. The preprocessing stage provides local image features, independent of particular image size, encoding format, scale and orientation, which is desirable for web images analysis. There are many such features [3, 21] which are often used in conjunction, but for getting an insight on the available data (and for practical reasons of working with a huge dataset) only one type of features is currently selected - colour SIFT [16, 26].

The first stage (Figure 1) creates a single representation for each image, which is required for the image classifier and does not depend on the particular number of local features. For the representation, a histogram of local features of each class is used. Local features are classified with a random vector quantization approach.

The second stage (Figure 1) estimates classes of single images. Any non-linear classifier which scales to millions of data samples suits for that role. An ELM [12] classifier is chosen for low computational requirements and a performance comparable to the state-of-the-art [17].

The last stage combines estimations of image classes into website classes. It is applied when image grouping into websites is available. The steps are described in more details in the following subsections.

### 2.1 Local Features

Local image features [15] are widely used as a base in image processing systems, including the state-of-the-art ones [6]. They can robustly capture similar objects in different images under slightly different angles or poses [24], and are tolerant to image scaling, rotation and noise caused, for instance, by different encodings and compression [16].

Local image features are calculated in two steps: determining features' positions, sizes and orientations (*feature detection*), and calculating quantized histogram from the pixel values of an image patch (*feature description*).

Several algorithms exist for feature detection [24], generally with a trade-off between detection accuracy and invariance to image transformations. The Harris-Laplace [18] feature detector is used in the work, which is rotation and scale invariant. It starts from building edge maps of the intensity map of the whole image, for different scale parameter. This gives a 3-dimensional tensor, local maximums of which corresponds to the detected local features. Orientation is just the major orientation of pixel gradients in a feature region.

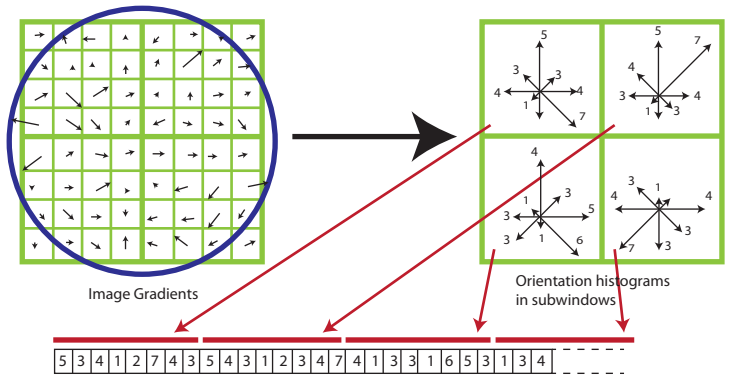



**Fig. 2.** Schematic representation of local feature descriptor calculation. Real descriptor has 4×4 sub-windows instead of 2×2 on the picture.

The selected feature description method is based on a widely used SIFT [19]. It calculates the descriptors by splitting the detected image patch into 4×4 bins

(2×2 on the figure), quantizing pixel gradients in each bin, and concatenating the quantized representations. The process is illustrated on the Figure 2. The approach utilized in the paper is cSIFT [26], which calculates SIFT descriptors for each colour channel in the opponent colour space and concatenate them together. This allows automatically consider the colour of an image for colour-sensitive classification tasks like adult image detection.

### 2.2 Image Representation with Random Vector Quantization

A unified representation for all images is desirable for the classification task, however different images have highly variable number of local features (from one to tens of thousands). A typical approach of building a local feature histogram (also called a *bag-of-visual-words*) is omitted, as it aims at using an SVM classifier which has effective kernels for histogram data, and the classifier in the article is not an SVM.

The idea for image representation is to calculate how many local features of each class an image possesses. This corresponds to an *image-to-class distance* [4], and enables classification of a query image even if no similar image exists in the training set. Local features are classified using random vector quantization (random VQ) [10] of a cSIFT descriptors space, as described hereafter.

From all the training images, local descriptors $\mathbf{d}_i \in \mathbb{R}^{384}, i \in [\![1, N_d]\!]$ are gathered in the same pool $\mathcal{D} : \mathbf{d}_i \in \mathcal{D}$. Assume that all the descriptors in an image have the same class as an image itself $c_i = \text{class}(\mathbf{d}_i)$. Given a number of clusters $K$, the same number $K$ of descriptors $\mathbf{d}_k \in \mathcal{D}, k \in [\![1, K]\!]$ are chosen randomly from the whole pool of descriptors $\mathcal{D}$. Lets call the selected descriptors *centroids* $\mathbf{l}_k = \mathbf{d}_k, k \in [\![1, K]\!]$ for convenience. Then the set of centroids $\{\mathbf{l}_k\}, k \in [\![1, M]\!]$ defines a random VQ of the descriptor space, with the corresponding Voronoi cells $V_k : \forall \mathbf{d} \in V_k,\ k = \text{argmin}_{j \in [\![1,K]\!]} ||\mathbf{d} - \mathbf{l}_j||_2$. Denote $\mathcal{D}_k$ a set of descriptors belonging to the Voronoi cell $V_k$, and all possible classes $m \in [\![1, M]\!]$. Then classes can be assigned to the Voronoi cells (and the corresponding centroids) by the majority vote of classes of its descriptors:

$$c_k = \text{argmax}_{m \in [\![1,M]\!]} \sum_{\mathbf{d}_j \in \mathcal{D}_k} \delta(c_j, m),\ c_k = \text{class}(\mathbf{l}_k) = \text{class}(V_k), \qquad (1)$$

where $\delta(c_j, m)$ is Kronecker delta.

Better results are achieved using soft class assignment with $\mathbf{c}_k = [c_k^1 \dots c_k^M] \in \mathbb{R}^M$, provided by the empirical formula on eq. 2. A likelihood of a centroid $\mathbf{l}_k$ to belong to class $m$ depends on an average distance from the centroid (the smaller the higher) which is further divided by a number of descriptors of that class in the given Voronoi cell (the more the higher is the likelihood). The square root smooths out extreme likelihood values. If no descriptors of that class are presenting in that Voronoi cell, $c_k^m = -\sqrt{\text{global average distance}}$ is used.

$$c_k^m = -\sqrt{\sum_{(\mathbf{d}_i \in \mathcal{D}_k) \cap (c_i = m)} ||\mathbf{l}_k - \mathbf{d}_i||_2 \ / \ ||\{\mathbf{d}_i\}||}, \qquad (2)$$

Defining a set of descriptor in a query image $\mathcal{D}_q$, a representation of the query image is a vector $\mathbf{q} = [q^1 \dots q^m] \in \mathbb{R}^M$ is just a summation of soft class vectors for all of its descriptors:

$$\mathbf{q} = \sum_{\mathbf{d}_i \in \mathcal{D}_q} \mathbf{c}_k : k = \text{argmin}_{j \in [\![1,M]\!]} ||\mathbf{d}_i - \mathbf{l}_j||_2 \tag{3}$$

Choosing the class($\mathbf{q}$) = $\text{argmax}_{m \in [\![1,M]\!]}\, q^m$ already provides a naive image classification, however adding the proper classifier in the last stage significantly improves the results.

### 2.3 ELM classifier

The ELM[12, 11] algorithm uses the Single Layer Feedforward Network (SLFN) structure. The main concept behind the ELM is the random initialization of the weights and biases of the hidden layer of SLFN. Then finding an output weights is a linear problem, with a very low computational cost compared to iterative training procedure like error back-propagation.

Consider a set of $M$ distinct samples $(\mathbf{x}_i \in \mathbb{R}^{d_1}, \mathbf{y}_i \in \mathbb{R}^{d_2})$; then a SLFN with $N$ hidden neurons is modelled as the following sum

$$\sum_{i=1}^{N} \beta_i \phi(\mathbf{w}_i \mathbf{x}_j + b_i),\ j \in [\![1, M]\!] \tag{4}$$

with $\phi$ being the activation function, $\mathbf{w}_i$ the input weights, $b_i$ the biases and $\beta_i$ the output weights.

In the case where SLFN would perfectly approximate the data, the error between the estimated outputs $\hat{\mathbf{y}}_i$ and the actual outputs $\mathbf{y}_i$ are zero, and the relation between inputs, weights and outputs is then

$$\sum_{i=1}^{N} \beta_i \phi(\mathbf{w}_i \mathbf{x}_j + b_i) = \mathbf{y}_j,\ j \in [\![1, M]\!], \tag{5}$$

which writes compactly as $\mathbf{H}\beta = \mathbf{Y}$, with

$$\mathbf{H} = \begin{pmatrix} \phi(w_1 x_1 + b_1) & \cdots & \phi(w_N x_1 + b_N) \\ \vdots & \ddots & \vdots \\ \phi(w_1 x_M + b_1) & \cdots & \phi(w_N x_M + b_N) \end{pmatrix} \tag{6}$$

and $\beta = (\beta_1^T \dots \beta_N^T)^T$, $\mathbf{Y} = (y_1^T \dots y_M^T)^T$.

Solving the output weights $\beta$ from the hidden layer output matrix $\mathbf{H}$ and target values is achieved through the use of a Moore-Penrose generalized inverse of the matrix $\mathbf{H}$, denoted as $\mathbf{H}^\dagger$. The ELM is proven to perform universal function approximation [12].

In the proposed methodology, ELM classifier further processes image representations $\mathbf{q}$ to improve results by non-linear transformation and considering relations between the components of $\mathbf{q}$. The inputs are image representation vectors $\mathbf{q} \in \mathbb{R}^K$, the desired outputs $\mathbf{y} \in \mathbb{R}^K$ have all the components equal -1 except for the true class component set to +1. The classifier outputs $\hat{\mathbf{y}} \in \mathbb{R}$, and the estimated class of an image is $\hat{c} = \text{argmax}_m(\hat{\mathbf{y}}^m)$.

### 2.4 Website Classification

In image-based website classification, additional information is provided by the grouping of images into websites. An estimated website class $\hat{c}_{ws}$ is obtained from its images as:

$$\hat{c}_{ws} = \text{argmax}_m \Big( \sum_{\hat{\mathbf{y}} \in \text{website}} \hat{\mathbf{y}}^m \Big) \tag{7}$$

Experiments show the prediction improvement from combining image outputs for website classification, up to twice higher accuracy for some classes. The next section presents application results of the proposed methodology to a real web images dataset, as well as to a subset of Caltech101 for comparison.

## 3 Experiments and Results

The methodology was tested on two datasets - one real web image dataset provided by F-Secure Corp.[4] (which cannot be published), and a subset of Caltech101 for comparison. The web image dataset has 11 classes of interest plus one "background" *Unknown* class representing all other webpages. It has training and test parts, which are taken from separate websites. Not all the images are relevant because true classes are known only for webpages; part of irrelevant images may reach 80% for classes like *Cults*. The former dataset consists of 12 classes from Caltech101, which have the largest amount of images. Parameters of websites are presented in Table 1. Caltech101 dataset does not have webpages, thus only image classification results were obtained. Web images set has both image classification and webpage classification results.

Main code was written in Python, and it includes python imaging library with multiprocessing module for parallel computations. Colour image descriptors are obtained using colorDescriptor software [26]. Centroids for random vector quantization are initialized from random descriptors of the web images training dataset and used in both experiments, because they were taken randomly, and their labels are re-calculated for each dataset separately.

### 3.1 Web image dataset

The classifier was trained on the web images training set, using 10000 images per class for training and 7000-10000 images per class for validation. Experiments

[4] F-Secure Corporation, http://www.f-secure.com/en/web/home_global

**Table 1.** Datasets parameters, relevance estimated empirically.

| Web images dataset, training + test | | |
|---|---|---|
| Class | Websites | Images |
| Adult | 6801+219 | 216122+8597 |
| Alcohol | 12828+628 | 76666+5506 |
| Cults | 8387+182 | 37410+2262 |
| Dating | 4703+125 | 31844+789 |
| Drugs | 11439+221 | 85393+2727 |
| Gambling | 7322+582 | 38546+4905 |
| Hate | 8546+206 | 48581+1993 |
| Religion | 5438+45 | 17928+545 |
| Tobacco | 5784+483 | 38623+6379 |
| Violence | 1919+224 | 21113+4010 |
| Weapons | 2464+164 | 27240+1755 |
| Unknown | 3432+96 | 34382+902 |

| Caltech101 subset | |
|---|---|
| Class | Images |
| Aeroplanes | 800 |
| Motorbikes | 798 |
| Background | 468 |
| Faces easy | 435 |
| Faces | 435 |
| Watch | 239 |
| Leopards | 200 |
| Bonsai | 128 |
| Car side | 123 |
| Ketch | 114 |
| Chandelier | 107 |
| Hawksbill | 100 |

repeated a number of times with random initializations, and the best model selected with validation. Then the test results are calculated on the separate test set. The results are presented on Figure 3. The calculations were performed using computer resources within the Aalto University School of Science "Science-IT" project. Average running time per image was 0.3s for extracting the cSIFT descriptors (on average 300 descriptors per image) and 0.5s for finding the closest centroid for all the descriptors. Running the ELM for all 40000 images of the test set took 0.7s.

Single image classification provides 23,5% average accuracy (the random guessing would give 8,3%). Combining images to websites and calculating classes of websites increase average accuracy to 33,6%. Note that the combination step has increased accuracy for every single class, and for the *Drugs* class it increases twofold: from 21,6% to 43,4%.

### 3.2 Caltech101 subset

For the subset of Caltech101, only image classification accuracy is obtained, as there are no websites. For each class, 40 images are used for training, 30 for validation and 30 for testing, chosen randomly without repetitions. The best ELM classifier is chosen among 100 runs, and the whole process repeated 100 times. Results are presented on Figure 4. Average accuracy for test images is 52,5%.

## 4 Conclusion

A general image-based website classification methodology is presented in the paper. It works with arbitrary classes, tolerant to noise, and scales well with the size of dataset.

On real image classification test, it shows 23,5% average accuracy on single images (random guessing would give 8,3%), boosted to 33,6% by combining

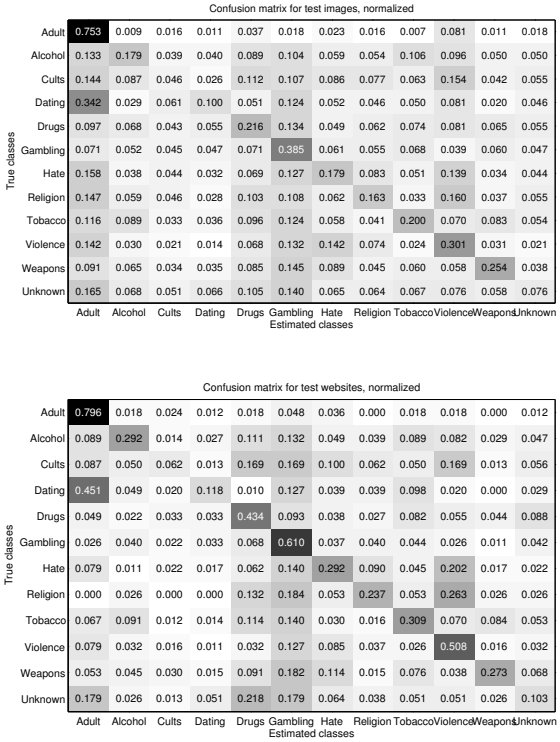



**Fig. 3.** Test results for web images dataset. Bottom chart shows results combined into webpages - collections of images. Combining results increases accuracy.

images into websites. Some classes, like *Adult* and *Dating* or *Cults* and *Religion*, are confused with each other, which reflects the true distribution of images. Overall, the classifier is working, and for some categories it manages to achieve very good performance.

The Caltech101 subset yield 52,5% average image classification accuracy. For a comparison, SVM+KNN method [28] report 56%-66,2% average accuracy for the whole Caltech101 image set, which is superior to the proposed algorithm. While on the current stage, the methodology does not compete to the state-of-the-art ones, the goal of testing in a Caltech101 subset was to check the results in the absence of noise. The increase of average accuracy from 23,5% to 52,5% with the same amount of classes shows that the noisy data problem is important, and encourages the development of image set filtering algorithms.

There are many ways of further development, which are easier to implement now, when the general framework for image-based website classification is ready. Images should be filtered using a separated advertisement and avatar image classifier, and the amount of text in image. One way to improve results is by using

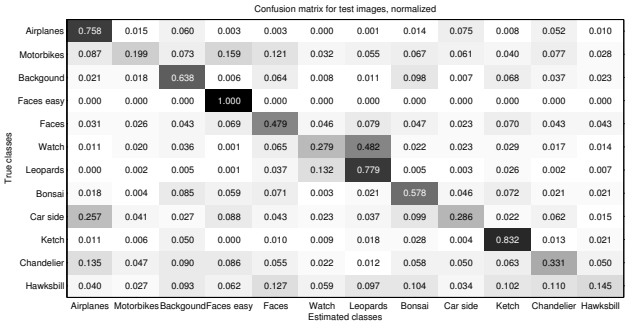



**Fig. 4.** Caltech101 subset classification test results, averaged over 100 runs.

a proper SIFT descriptors distance [22] instead of Euclidean norm. Second is to extract possible objects from images (which can be performed automatically [5]) and use them in classification. One more way is to add more local and global descriptors [2], such as GIST [21] or LBP [1]. Different global approaches are described in *Torralba*'s works [21, 23]. And the final step may include multi-view learning [25] from the aforementioned different sources. Such improvements are currently under investigation, and will be part of a future work.

## References


1. T. Ahonen. Face description with local binary patterns: Application to face recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.
2. L. Ballan and M. Bertini. Combining generative and discriminative models for classifying social images from 101 object categories. *21st International Conference on Pattern Recognition (ICPR)*, (September):1–4, 2012.
3. H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *Proc. of the 9th European conference on Computer Vision*, pages 404–417, 2006.
4. O. Boiman, E. Shechtman, and M. Irani. In defense of Nearest-Neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.*, number I, pages 1–8. IEEE, June 2008.
5. J. Carreira and C. Sminchisescu. Cpmc: Automatic object segmentation using constrained parametric min-cuts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(7):1312–1328, 2012.
6. Q. Chen, Z. Song, and Y. Hua. Hierarchical matching with side information for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3426–3433. IEEE, June 2012.
7. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.
8. L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

9. R. Fergus and L. Fei-Fei. Learning object categories from internet image searches. *Proc. of the IEEE*, 98(8):1453–1466, 2010.
10. A. Gersho and R. M. Gray. *Vector quantization and signal compression*. Springer, 1992.
11. G.-B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE trans. on systems, man, and cybernetics.*, 42(2):513–529, Apr. 2012.
12. G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1):489–501, Dec. 2006.
13. Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. *Pattern Analysis and Machine Intelligence*, 30(11):1877–1890, 2008.
14. P. Y. Lee, S. C. Hui, and A. C. M. Fong. Neural networks for web content filtering. *Intelligent Systems, IEEE*, 17(5):48–57, 2002.
15. D. Lowe. Object recognition from local scale-invariant features. In J. Tsotsos, editor, *Proc. of the Seventh IEEE International Conference on Computer Vision*, volume 2 of *ICCV '99*, pages 1150–1157. Kerkyra, Greece, IEEE, 1999.
16. D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.
17. Y. Miche, M. van Heeswijk, P. Bas, O. Simula, and A. Lendasse. TROP-ELM: A double-regularized ELM using LARS and Tikhonov regularization. *Neurocomputing*, 74(16):2413–2421, Sept. 2011.
18. K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. *ICCV 2001.*, 1:525–531, 2001.
19. K. Mikolajczyk and C. Schmid. Performance evaluation of local descriptors. *IEEE trans. on pattern analysis and machine intelligence*, 27(10):1615–1630, Oct. 2005.
20. M. Mofaddel and S. Sadek. Adult image content filtering: A statistical method based on Multi-Color Skin Modeling. In *2010 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 682–686, 2010.
21. A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3), 2001.
22. O. Pele and M. Werman. The quadratic-chi histogram distance family. *Computer Vision - ECCV 2010*, (1):1–14, 2010.
23. A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE trans. on pattern analysis and machine intelligence*, 29(5):854–69, May 2007.
24. T. Tuytelaars and K. Mikolajczyk. Local Invariant Feature Detectors: A Survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.
25. G. Tzortzis and A. Likas. Convex mixture models for multi-view clustering. *Artificial Neural NetworksICANN 2009*, pages 205–214, 2009.
26. K. van de Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE trans. on pattern analysis and machine intelligence*, 32(9):1582–1596, Sept. 2010.
27. V. Viitaniemi. *Visual category detection: an experimental perspective*. PhD thesis, 2012.
28. H. Zhang and A. Berg. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:2126—-2136, 2006.
29. Q. Zheng, W. Zeng, W. Wang, and W. Gao. Shape-based adult image detection. *International Journal of Image and Graphics*, 6(1):115–124, 2006.

# Combining Multiple Classifiers and Context Information for Detecting Objects under Real-world Occlusion Patterns

Marvin Struwe[1], Stephan Hasler[2], and Ute Bauer-Wersing[1]

[1] University of Applied Sciences Frankfurt am Main, Germany
[2] Honda Research Institute Europe GmbH, Offenbach, Germany
{mstruwe,ubauer}@fb2.fh-frankfurt.de
stephan.hasler@honda-ri.de

**Abstract.** Current state-of-the-art detection approaches reveal a strong performance degradation with increasing object occlusion. Here we investigate different strategies to improve detection of occluded objects based on the analytic feature framework from [8] and compare the results in a car detection task. Motivated by an analysis of annotated traffic scenes we first describe a dedicated combination of classifiers to deal with the predominant car-car occlusion, and second, we propose a more general concept to handle vertical occlusion patterns. In a final test, depth information is used as additional local cue to reason about visible object parts. We report first improvements and discuss advantages and drawbacks of the individual approaches for further investigations.

**Keywords:** Object detection, Occlusion handling, Supervised learning

## 1 Introduction

Despite extensive research visual detection of objects in natural scenes is still not robustly solved. The reason for this is the large appearance variation in which objects or classes occur. A very challenging variation is occlusion which is caused by the constellation of objects in a scene. Occlusion reduces the number of visible features of an object but also causes accidental features. Existing object representations can deal moderately well with a low to medium level of occlusion and fail for stronger occlusions. The parts-based methods like [6, 7] aggregate local features in a voting manner and are usually trained with unoccluded views. During recognition they can handle arbitrary occlusion patterns, but require that sufficiently many features can still be detected. In contrast to this other methods like [1, 8] train a holistic object template in a discriminative manner. These methods focus resources on differences between classes. Because of this strong specialization on the training problem, these approaches show a stronger decrease of performance for occluded objects when trained on unoccluded views. However, in general the voting methods perform worse than the discriminative ones, whenever test and training set do not show such systematic

differences, as discussed in [13] and confirmed by the detection results in [2]. In this paper we investigate different strategies to improve the performance of a holistic discriminative method under stronger occlusion.

A common strategy to explicitly deal with occlusion is to make use of context information, i.e. to exploit knowledge about the possible constellation of objects. In [3, 12] Markov-Random-Fields are used to infer if neighboring features are consistent with a single detected instance of an object or have to be assigned to different ones. In this way both approaches can reason about relative depth of objects and produce a coarse segmentation. However, both methods require a time consuming iteration process over the whole input image. In [10] spatial relations are exploited more selectively by using the detections of larger and thus more easily detectable objects to narrow down search space for smaller, more difficult ones. We transfer this general concept to the occlusion case, where the difficult objects are the occluded ones.

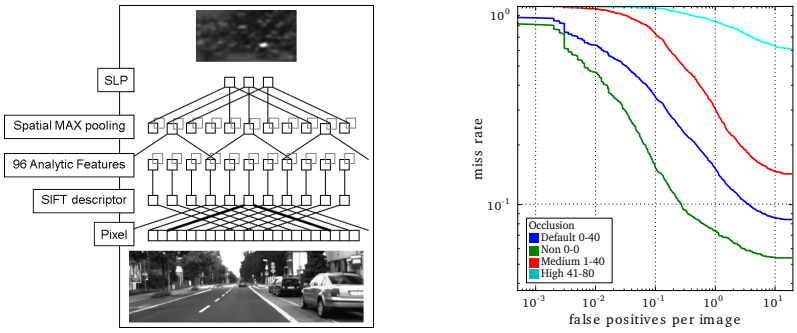
Besides instance-instance relations also knowledge about general occlusion patterns can be used. In [11] the authors explicitly take vertical occlusion at the image border into account. When objects are placed on a horizontal plane, vertical occlusion is a typical pattern also inside the image. Here we exploit this for car detection by using a dedicated classifier architecture.

Occlusion is related to the 3D relation of objects. A general cue of 3D information is depth. To check the physical plausibility of an object's position and size [4] or to segment and put attention to individual scene elements. So in [9] temporal differences between RGB-D(epth) views are used to discover movable parts for action representation. Here we integrate depth into the architecture for vertical occlusion to reason about visibility of features.

In Sec. 2 we shortly describe our basic holistic discriminative detection framework and show how it is influenced by occlusion in a car detection task. In the following sections different strategies to deal with occlusion are motivated and tested. First, in Sec. 3 we propose a conditional combination of classifiers for the predominant car-car occlusion setting. To deal with more general occlusion pattern we design a classifier architecture in Sec. 4 where the response of discriminative vertical parts detectors are integrated in a second stage. Finally, in Sec. 5 we make a first test how to exploit depth information in the 2-stage architecture, before drawing the conclusion in Sec. 6.

## 2 Analytic Feature Framework

Holistic discriminative approaches usually extract unspecific features and apply a powerful classifier directly on top. So the popular method proposed in [1] uses Histograms of Oriented Gradients (HOG) with a Support Vector Machine (SVM) and was shown to yield state-of-the-art performance in various detection tasks. In [8] we proposed the analytic feature framework (Fig. 1a) that puts effort in learning a more problem-specific feature representation and uses a simple classifier on top for discrimination. We could show that it provides competitive detection performance. For an input image first SIFT descriptors are computed




(a) (b)

Fig. 1: (a) Analytic feature hierarchy. SIFT descriptors are computed on a regular grid and matched to 96 analytic features. After a local maximum filter per feature the SLP templates are used in a convolutional step. Maxima in the final response map denote possible car locations. (b) ROC of $C_{\text{Std}}$ for car detection scenario. The performance decreases strongly with the percentage of the cars' occlusion.

on a regular grid and then matched to a set of 96 analytic features. The analytic features are the result of a supervised selection process described in [5]. Next, a local maximum filter is performed per feature to enhance robustness against small translations. Finally the car template, which was trained with a Single Layer Perceptron (SLP), is shifted over the feature representation. The local maxima in the resulting response map denote possible car locations. To deal with cars at different distances we apply the framework on successively reduced image resolutions. For a pedestrian benchmark in [8] we could prove highly competitive performance of the analytic feature framework approach.

In [8] we used the framework for detection of front and back views of cars in real world traffic scenes. These image streams were taken under different weather conditions (sunny, rainy, overcast) and in different scene types (city, rural, industry, highway) and contained cars under all levels of occlusion. The final SLP car template was trained on unoccluded views only. We will refer to this reference system as $C_{\text{Std}}$ throughout the paper.

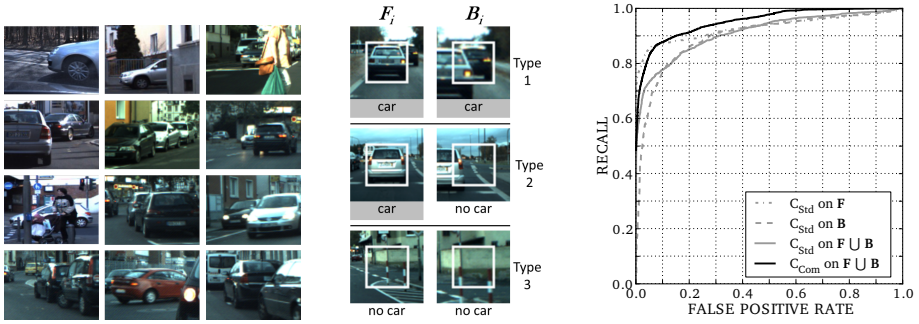
The results in Fig. 1b reveal a strong dependency of the performance of $C_{\text{Std}}$ on the percentage of the cars' occlusion. For a false positive per image rate of 0.1 we get 70% of the cars with an occlusion between 0-40%. This pure detection performance is usually sufficient for a system that applies temporal integration (tracking). However for stronger occlusion the recall drops severely, which can no longer be compensated at system level.

## 3 Occlusion Handling Using Object-Object Relations

To get a better understanding of occlusion of cars in traffic scenes we counted the number of typical occluders and types of occlusion using ground truth data

Table 1: Counts of car occluders and occluded car parts for the ground truth data. In total 8796 out of 15514 cars are occluded, most of them by other cars.

| Occluding object | # | Occluding object | # | Occluded part | # |
|---|---|---|---|---|---|
| Another car | 7061 | Pedestrian | 70 | Left | 3730 |
| Image border | 2137 | Traffic sign | 31 | Right | 3124 |
| Motor bike | 82 | Other/non-labeled | 1125 | Middle (only) | 90 |




(a) (b) (c)

Fig. 2: (a) Typical occlusion examples. (b) Segment pair types. Each pair has a foreground segment $\boldsymbol{F}_i$ and a background segment $\boldsymbol{B}_i$. For $\boldsymbol{B}_i$ the first pair-type contains a labeled car occluded by another car, the second pair-type a random region next to a car, and the third pair-type a random region next to a false-positive of $C_{\text{Std}}$. For simplification we only use samples with occlusion at the left side and mirror examples with right occlusion to get more data. (c) $C_{\text{Std}}$ shows a good performance for the foreground segments $\boldsymbol{F}$ while the result for the occluded cars $\boldsymbol{B}$ is significantly weaker. On the combined data set $\boldsymbol{F} \cup \boldsymbol{B}$, $C_{\text{Com}}$ is in general much better than $C_{\text{Std}}$.

(see some occlusion examples in Fig. 2a). The result in Tab. 1 reveals that most cars are occluded by other cars. In [8] we exploited this fact and trained an additional classifier $C_{\text{Occ}}$ on occluded cars and applied in the vicinity of cars already detected by $C_{\text{Std}}$ only. This concept was inspired by [10] where the detection of small office objects was enhanced by predicting their spatial position relative to larger and more easily detectable objects.

We decided to use two separate classifiers instead of training $C_{\text{Std}}$ also with occluded views, because we expected a decrease of performance of $C_{\text{Std}}$ for unoccluded cars otherwise. The conditional application of $C_{\text{Occ}}$ is necessary to avoid a strong increase in the number of false positives, which would be the result of the independent usage of both classifiers. We refer to the combined application of both classifiers as $C_{\text{Com}}$.

For a fast proof of concept, in [8] we tested this strategy on segmented car and non-car views. So we generated data pairs $i$, each having a **F**oreground segment $\boldsymbol{F}_i$ containing the occluder and the corresponding **B**ackground segment $\boldsymbol{B}_i$ containing something occluded. We refer to the set of all foreground/background segments with $\boldsymbol{F} = \{\boldsymbol{F}_i\}$ and $\boldsymbol{B} = \{\boldsymbol{B}_i\}$ respectively. The types of pairs that mimic all possible constellations in a scene are shown in Fig. 2b. We trained $C_{\mathrm{Occ}}$ on the background segments $\boldsymbol{B}$ including cars with an occlusion up to 80%.

Figure 2c confirms again that $C_{\mathrm{Std}}$ can cope substantially better with the familiar foreground segments $\boldsymbol{F}$ than with the occluded segments in $\boldsymbol{B}$. On the combined data set $\boldsymbol{F} \cup \boldsymbol{B}$ the classification has some intermediate quality but is clearly dominated by $C_{\mathrm{Com}}$. For example, at a recall of 0.8 $C_{\mathrm{Std}}$ has a false positive rate of 0.13, while that of the combined curve is 0.04. This is a threefold reduction in the number of false positives.

The used segment dataset in [8] was simple in the way that the position and size of the occluded car was normalized, whereas in a real scene a strong variance can be expected relative to the position and size of the unoccluded car. Because of this it is not trivial to transfer the proposed concept to full scene detection. Furthermore the concept neglects general occluders. Thus we propose other new strategies to deal with occlusion in the following.

## 4 Split of the holistic car template

Tab. 1 reveals that most cars are either occluded on the right or left side. This vertical occlusion is caused by other cars, unlabeled walls, or the image border (see Fig. 2a), and causes a mismatch of the holistic car template used in $C_{\mathrm{Std}}$. To improve the detection for this type of occlusion we used following strategy: We subdivided the holistic classifier into three vertical parts and trained each part-classifier with unoccluded car views. So each classifier is forced to make a more local decision about the presence of the car and is later not affected by occlusion of a different part. To integrate the responses of the part-classifiers we use their confidence values as input for an additional SLP which is trained with cars with occlusion rate 0-80%. The resulting two stage architecture is shown in Fig. 3b and will be referred to as $C_{\mathrm{3Split}}$. The structure is equivalent to an MLP but instead of Backprop learning we use for each stage a different training set. In this way we enforce a stronger local decision and better generalization of each sub-segment.

The detection results in Fig. 4a show an improved performance of $C_{\mathrm{3Split}}$ compared to $C_{\mathrm{Std}}$ for occlusion rates of 1-40%, while there is a similar performance for occlusion rates of 41-80%. Unexpectedly, there is a strong gain for unoccluded cars. A possible reason might be that the vertical part templates of $C_{\mathrm{3Split}}$ are forced to make better use of their local information and thus find a more general car concept each. This hypothesis is underlined by the weights learned by the holistic SLP $C_{\mathrm{Std}}$, that show a rather sparse contribution of a small set of analytic features at specific locations, while each part-classifier of $C_{\mathrm{3Split}}$ integrates all features at all positions in a much broader manner.

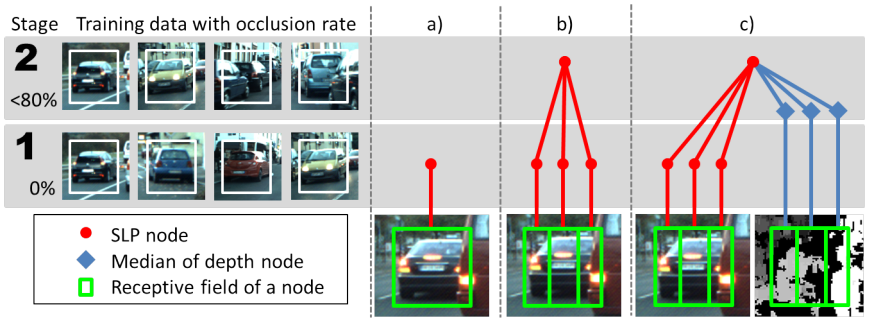


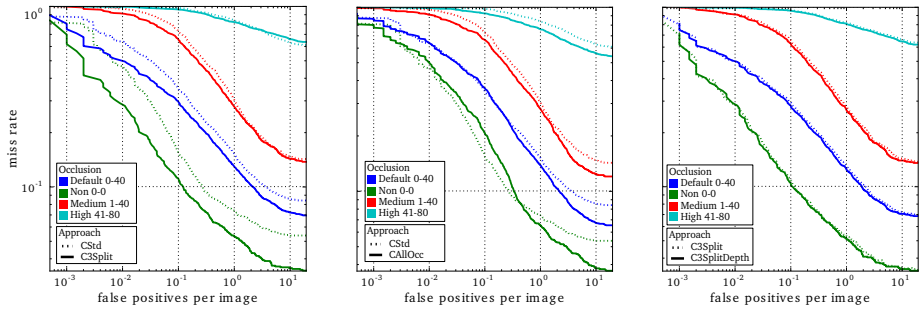

Fig. 3: Different detection architectures. (a) Architecture of $C_{\text{Std}}$. A holistic SLP template is trained on non-occluded car views. (b) Architecture of $C_{\text{3Split}}$. The three vertically subdivided SLP templates are trained on the same examples to (a). In the second stage, a separate SLP learns to combine the three confidence values of the first stage using non-occluded and occluded car views. (c) Architecture of $C_{\text{3SplitDepth}}$. The final SLP uses the median depth for each vertical car part as additional input.

Motivated by the result of $C_{\text{3Split}}$ we splitted each vertical part further into two horizontal regions. However, $C_{\text{6Split}}$ showed a much worse performance compared to $C_{\text{3Split}}$ in a similar range as $C_{\text{Std}}$. In future more systematic analysis is necessary to investigate the effects of this two stage architecture.

$C_{\text{3Split}}$ differs from $C_{\text{Std}}$ in two ways: It uses an adapted classifier architecture and additional occluded training examples in the second stage. To verify that the improved performance of $C_{\text{3Split}}$ is not simply caused by using different training data, we trained a holistic detector similar to $C_{\text{Std}}$ but using the training data of the second stage of $C_{\text{3Split}}$ with occlusion of 0-80%. We refer to this classifier as $C_{\text{AllOcc}}$. In Fig. 4b, $C_{\text{AllOcc}}$ shows a similar gain as $C_{\text{3Split}}$ for 1-40% occlusion. The performance for unoccluded cars is worse than $C_{\text{Std}}$. This indicates that the additional variation in the occluded training examples confuses the representation of unoccluded views, as we expected in Sec. 3. For 41-80% occlusion $C_{\text{AllOcc}}$ outperforms $C_{\text{3Split}}$ and $C_{\text{Std}}$, so the holistic approach can better make use of the remaining information in case of strong occlusion, maybe by directly representing the effect of the occlusion edge.

## 5 Additional use of depth information

$C_{\text{3Split}}$ shows an improvement in detection performance for cars with an occlusion rate of 0-40%. For cars with stronger occlusion the performance is nearly the same as $C_{\text{Std}}$. The SLP in the second stage seems not to be able to deal with some occlusion cases by using only the confidence values of the sub-segments. A first analysis of the results shows the following problem: Very low confidence values in two of three sub-segments can result in a low confidence value at the classifier on top regardless of how good the confidence value on the third




(a) (b) (c)

Fig. 4: (a) Performance of $C_{\text{3Split}}$. $C_{\text{3Split}}$ generally dominates $C_{\text{Std}}$ at occlusion rates up to 40%, with an unexpected, strong gain for unoccluded views. (b) Performance of $C_{\text{AllOcc}}$. $C_{\text{AllOcc}}$ shows a similar gain as $C_{\text{3Split}}$ for 1-40% occlusion. For unoccluded views the performance is worse than $C_{\text{Std}}$. But $C_{\text{AllOcc}}$ outperforms the other detectors on strongly occluded cars. (c) Performance of $C_{\text{3SplitDepth}}$. The way $C_{\text{3SplitDepth}}$ makes use of depth information does not lead to a significant improvement.

sub-segment is. There are two different scenarios where this constellation can happen. First, the object is a non-car object but a structure in one sub-segment accidentally generates a high confidence value, which can potentially happen quite often. Second, the object is a strongly occluded car and only visible in one of the three sub-segments. The approach is not able to distinguish both patterns and thus seems to require a strong response of two vertical parts at least. One possibility to overcome this limitation is to make use of additional context cues, e.g. stereo disparity. So the fact that the occluded part of a car will be closer to the camera than the visible part could be exploited.

We tested this hypothesis by retraining the SLP on the second stage of $C_{\text{3Split}}$ using the median depth of the three vertical car parts as additional input. The resulting architecture is shown in Fig. 3c and referred to as $C_{\text{3SplitDepth}}$. The results in Fig. 4c show no significant improvement compared to $C_{\text{3Split}}$, so the $C_{\text{3SplitDepth}}$ seems not to be able to make use of the additional cue. An analysis of the top-level SLP weights shows that the median depth information is nearly unused. There are several possible reasons for this. Maybe the use of the median per part is a too strong reduction of information, or the use of the absolute median values does not produce a linearly separable pattern for different distances of the car and its occluder. We need a detailed analysis of the depth information to find a useful representation which can increase the detection performance.

## 6 Conclusion

In this paper we presented different alternatives to improve the detection of occluded objects. In a first test we combined two detectors, one for unoccluded

and one for occluded instances, in a conditional way, and reported enhanced performance in a car-car occlusion setting using segmented views. For the more general setting of vertical occlusion we proposed a vertical split of the holistic car template into three parts together with a second stage that learns typical combination of these parts responses for occluded car examples. This prototype outperformed the reference system for different levels of occlusion. However, there was only a very small gain for strongly occluded cars, but an unexpectedly large gain for unoccluded ones. We confirmed that the gain in performance is not simply caused by the additional use of occluded training examples and we also shortly tested an additional horizontal split of the vertical parts, which strongly reduced the performance. Finally, we integrated depth information into the vertical occlusion prototype to give the classifier an independent cue to reason about the response patterns of the vertical parts. However, the straightforward use of the median depth per part seems to be too coarse or inadequate to get a significant performance gain. In conclusion, the experiments already suggest some possible directions for further improvement. But much more in-depth analysis of the individual effects presented in this paper are necessary.

## References


1. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. Proc. CVPR (2005) 886–893
2. Dollar, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian Detection: A Benchmark. Proc. CVPR (2009) 304–311
3. Gao, T., Packer, B., Koller, D.: A Segmentation-aware Object Detection Model with Occlusion Handling. Proc. CVPR (2011) 1361–1368
4. Gould, S., Baumstarck, P., Quigley, M., Y. Ng, A., Koller, D.: Integrating Visual and Range Data for Robotic Object Detection. ECCV workshop M2SFA2 (2008)
5. Hasler, S., Wersing, H., Kirstein, S., Körner, E.: Large-scale Real-time Object Identification Based on Analytic Features. Proc. ICANN (2009) 663–672
6. Leibe, B., Schiele, B.: Interleaved Object Categorization and Segmentation. Proc. BMVC (2003) 759-768
7. Lowe, D. G.: Distinctive Image Features from Scale-invariant Keypoints. In: IJCV 60(2) (2004) 91–110
8. Struwe, M., Hasler, S., Bauer-Wersing, U.: Using Analytic Feature for the Destection of Occluded Objects. To appear in ICANN (2013)
9. Stückler, J., Behnke, S.: Hierarchical Object Discovery and Dense Modelling From Motion Cues in RGB-D Video. To appear in IJCAI (2013)
10. Torralba, A., Murphy, K. P., Freeman, W. T.: Contextual Models for Object Detection Using Boosted Random Fields. Proc. ICIP (2011) 653–656
11. Vedaldi, A.,Zisserman, A.: Structured Output Regression for Detection with Partial Truncation. Proc. NIPS (2009)
12. Winn, J., Shotton, J. D. J.: The Layout Consistent Random Field for Recognizing and Segmenting Partially Occluded Objects. Proc. CVPR (2006) 37–44
13. Yi-Hsin, L., Tz-Huan, H., Tsai, A., Wen-Kai, L., Jui-Yang, T., Yung-Yu, C.: Pedestrian Detection in Images by Integrating Heterogeneous Detectors. IEEE Computer Symposium (ICS) (2010) 252–257

# Anticipating Intentions as Gestalt Formation: A Model Based on Neural Competition

Martin Meier, Robert Haschke and Helge J. Ritter *

Neuroinformatics Group
Bielefeld University, 33501 Bielefeld, Germany
{mmeier,rhaschke,helge}@techfak.uni-bielefeld.de

**Abstract.** Anticipating the intentions of others is a key ability for cognitive interaction that is still not well understood and poorly replicated in artificial systems, such as robots. In this contribution we explore a neural model of Gestalt formation as a potential approach to intention anticipation. The idea is to view the already recognizable part of an ongoing action, together with the underlying intention, as a "Gestalt", which has to be completed when only the recognizable action part is given as an available fragment. To test this idea, we extend a previously developed model of competing neural layers for Gestalt formation by a "hallucination mechanism" that constructs the most likely completion of a given action fragment. We show that the resulting model can successfully anticipate cooperative moves of a human player in a two-person interaction scenario.

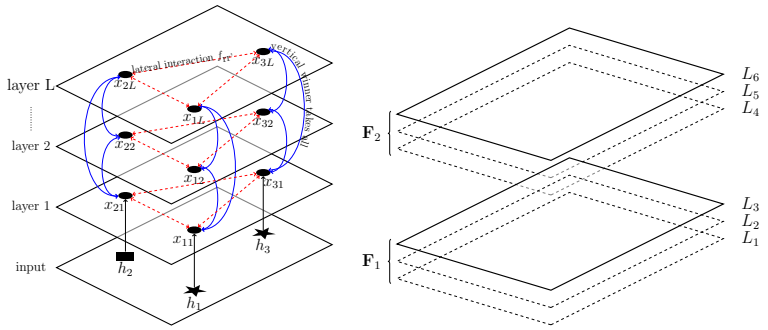
## 1 Introduction

When humans try to solve a task cooperatively, they can adapt their behavior to each other without explicit negotiation. This ability has many facets, one of them being the intuitive anticipation of the intentions of their collaboration partners. As an example, one can look at the task waiters perform when laying out dishes for a dinner. If one waiter starts to lay out the dishes, another one will most likely start to place the cutlery, because the dishes are taken care of and the cutlery is needed to complete the task.

This ability of "seeing" what the other will do to complete my fragmentary action has some resemblance to the completion of a fragmentary "Gestalt", viewing the fragmentary action as the incomplete pattern for which a "good continuation" is sought. This suggests to apply models for Gestalt formation to map the given input into a dynamics that completes the fragmentary actions towards a "good Gestalt", such as, e.g. "cooperation". The driving dynamics itself would then be in the role of the "intention" that completes the action.

To explore this idea in a concrete fashion, we apply the Competitive Layer Model [11]. The CLM has been proven feasible for a large set of segmentation

* This work has been conducted within and funded by the German collaborative research center "SFB 673: Alignment in Communication" granted by DFG.




(a) CLM (b) Multiple layer classes

**Fig. 1.** The left image shows the Competitive Layer Model. At the bottom are three inputs $v_{1\ldots3}$ and their corresponding neurons $x_{1\ldots3}$ within each layer $L = 1 \ldots \alpha$. The right image shows an example for a CLM with six layers and two different classes of interaction functions. The layers $L_1, \ldots, L_3$ respond to interaction function $\mathbf{F}_1$ and the layers $L_4, \ldots, L_6$ respond to interaction function $\mathbf{F}_2$.

and grouping problems in the image processing domain [9,14] and for automatic task segmentation [10].

The CLM allows the grouping of features based on the *Laws of Gestalt Theory* [6], a theory from the field of cognitive psychology, which tries to describe how humans perceive complex scenes. As a result of a combination of excitatory and inhibitory couplings of neurons, similar features form reinforcing groups of attractors, while simultaneously suppressing other, less similar features.

However, a human can not only correctly group features, but also imagine well fitting completions. Hence, we apply the approach presented in [7], with which it is possible to evaluate the compatibility of previously unknown features with respect to a CLM grouping result. The presented approach is extended with a technique to automatically find well fitting completions.

The recognition of intentions is of particular interest in the field of human-robot interaction and has been approached with a variety of techniques. In [5], the recognition of intention was done based on Hidden Markov Models, whereas [3] integrates Markov Models, Bayesian Networks and machine learning techniques in a hierarchical model to achieve this goal. Encoding a set of possible intentions in a Finite State Machine and learning of the transition probabilities has been done in [1].

## 2 The Competitive Layer Model

The Competitive Layer Model *(CLM)* is a recurrent neural network which consists of $L \times N$ linear threshold neurons. These neurons are arranged in $\alpha = 1 \ldots L$ layers, where each layer holds a total number of $r = 1 \ldots N$ neurons, as depicted in Fig. 1(a). The activity of a neuron in a single layer is denoted as $x_{r\alpha}$. The dynamics is designed such, that compatible features induce high neuron activities

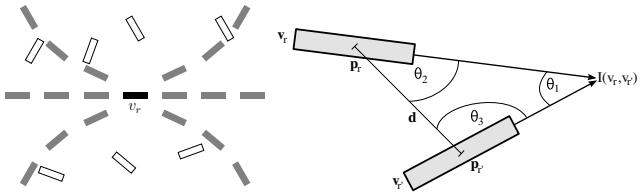



**Fig. 2.** An exemplary interaction function for good continuation is shown on the left. The central feature $v_r$ gets excitatory responses from the gray shaded features and inhibitory responses from the white. The right image shows parameters for the compatibility of oriented edge elements. The used parameters are the distance between the centers of the elements, given by $p_r$ and $p_{r'}$ and the three angles $\theta_{1,2,3}$.

within individual layers $\alpha$, indicating perceptual grouping of those features. All neurons within a column $r$ correspond to the same feature $v_r$.

The neurons in each layer are coupled with lateral interaction weights $f_{rr'}$, which are determined by the similarity between two features $v_r$ and $v_{r'}$, e.g. positive values for compatible and negative values for dissimilar features. This compatibility measure needs to be explicitly specified by a symmetric interaction function

$$f_{rr'} = \mathbf{f}(v_r, v_{r'}) = \mathbf{f}(v_{r'}, v_r). \tag{1}$$

Because the interaction weights need only be computed once, they can be stored as a symmetric interaction matrix $\mathbf{F} = f_{rr'}$.

An example for an interaction function that models the Gestalt Law of good continuation is given in Fig. 2(a). Features which form a smooth path with respect to the central feature $v_r$ create excitatory responses while other features which do not fit create inhibitory responses. To assure that an input feature $v_r$ is only represented by an active neuron in a single layer, the corresponding neurons in each layer are coupled with a columnar *winner-takes-all (WTA)* competition.

Combining both components, the lateral interaction function and the columnwise *WTA* circuit, a linear threshold dynamics can be summarized with the following update rule:

$$\dot{x}_{r\alpha} = -x_{r\alpha} + \sigma(J(h_r - \sum_{\beta} x_{r\beta}) + \sum_{r'} f_{rr'} x_{r'\alpha}) \tag{2}$$

Here, $\sigma(x) = \max(0, x)$ is the linear threshold function, $J(h_r - \sum_{\beta} x_{r\beta})$ represents the columnar *WTA* with a weighting constant $J$, and $h$ specifies the overall columnar activity and thus denotes the importance of a feature. Throughout this article we assume that all features are equally important, therefore $h = 1$. The lateral interaction between the features at position $r$ and $r'$ is computed with $\sum_{r'} f_{rr'} x_{r'\alpha}$. For a conclusive analysis of the CLM dynamics please refer to [14].

The CLM architecture as described above can only hold one interaction function at a time and therefore only respond to a specific type of feature compatibility. To circumvent this limitation, we apply the idea presented in [12] and introduce different types of layer classes. As shown in the example in Fig. 1(b),

a set of layers is grouped in a layer class which responds to a class specific interaction function. In the illustration in Fig. 1(b) the layers $L_1, \ldots, L_3$ respond to the interaction function $\mathbf{F}_1$ while the layers $L_4, \ldots, L_6$ respond to $\mathbf{F}_2$.

In previous works, for example in [14], the interaction function had to be hand crafted. To gain a better generalization capability and to simplify the task of choosing a suitable interaction function, we apply the techniques presented in [13] to learn an interaction function from labeled training data.

The approach presented in [13] relies on a set of labeled training data which is used twofold. In the first step, for each feature pair $(v_r, v_{r'})$ a proximity vector $d(v_r, v_{r'})$ is calculated. These proximity vectors are subsequently clustered to yield a compact representation of the proximity space. In the simplest case, the proximity vector may simply stack the original feature vectors, i.e. $d_{rr'} = [v_r^t, v_{r'}^t]^t$. However, if more insight into the task is available, more elaborate distance vectors can be computed, expressing the similarity of two features within a multi-dimensional vector. The objective is to capture properties of typical feature pairs within the given domain. For example, Fig. 2(b) shows the proximity function employed for oriented edge features.

In a second step, the label information is used to assign positive or negative interaction weights to each cluster prototype. Prototypes mainly corresponding to feature pairs of the same group, i.e. being compatible, will get a positive weight. Contrarily, pairs of features originating from different groups will generate negative interaction weights. Exploiting the frequency of the positive and negative interaction labels, it is possible to create a set of basis interaction functions. For a detailed derivation please refer to [13].

## 3 "Intention Field" for Anticipating new Features

To extend the capabilities of the CLM from grouping towards amending sparse groups, we utilize an approach presented in [7]. Given a CLM which has already converged to a grouping result, it is possible to use this result to find previously unknown features which fit well to the already known features.

### 3.1 Quality Measurement

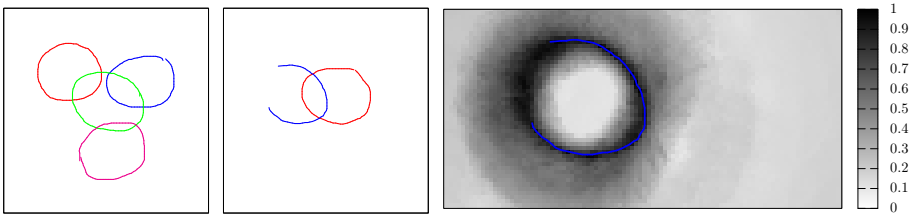
As presented in [7], it is possible to evaluate the quality $Q$ of a new feature $v_{new}$ by calculating its compatibility to the achieved grouping result. To this end, first an interaction vector $\mathbf{m}$ for the new feature value $v_{new}$ has to be created

$$\mathbf{m} = (f(v_{new}, v_0), f(v_{new}, v_1), \cdots, f(v_{new}, v_r))^T \tag{3}$$

utilizing the previously learned interaction function $f_{rr'}$. The support for the new feature is calculated as

$$x_{v_{new}\alpha} = \mathbf{m}^T \cdot \boldsymbol{x}_\alpha \tag{4}$$

for all layers $L = 1 \ldots \alpha$. We will view the mapping $I : (v, v_{new}) \rightarrow x_{v_{new}}$ from the existing pattern $v$ to the support as the "intention" $I(v, .)$ that is associated with




(a) Training data. (b) Grouping result for test data. (c) Activities from "intended" completions for the grouping result from (b).

**Fig. 3.** The labeled training input to learn the interaction function is shown in 3(a). The shapes are composed of edge features as depicted in Fig. 2(b). 3(b) presents the grouping result from the CLM for two test shapes. Their "intended" completions are displayed in 3(c), where the quality of the "intentions" produces a steep slope inside and outside of the circular input shapes.

the existing pattern $v$. In this way, we obtain a computationally clear and concise representation of "intention" as a mapping that tells how strongly the system supports different possible completions $v_{new}$ of a given partial input. To make the computed activity for the "intended" completing feature $v_{new}$ comparable, we apply a normalization, which shall assure that the support for such a feature is in the range from 0 to 1 in each layer.

Assume we have a CLM which holds $N$ features and is already converged. Because of the linear threshold $\sigma(x)$ from (2), every feature has an activity greater or equal to zero. Therefore, we calculate the normalization constant $M_\alpha$ for each layer as:

$$M_\alpha = \sum_{r=1}^{N} x_{r\alpha} \tag{5}$$

With $N$ being the number of features in the original input and their corresponding neural activity $x_{r\alpha}$ in the layer $\alpha$. Consequently, the quality $Q$ of an "intended" completion can be determined using (4) and (5) as

$$Q_\alpha(v_{new}) = \frac{1}{M_\alpha} \cdot \mathbf{m}^T \cdot \boldsymbol{x}_\alpha \tag{6}$$

An example of this process is shown in Fig. 3. An interaction function is learned from the training set in Fig. 3(a), where the labels are indicated by different colors. These shapes are composed of oriented edge features as shown in Fig. 2(b). The grouping result of the CLM is shown in Fig. 3(b). Here the assignment of features to the same layer is expressed by the same color. To closer examine the approach, the best fitting "intended" completions for the incomplete shape are generated. Fig. 3(c) shows the maximal quality of "intended" completions for each pixel of the image. Please note that for visualization purposes the feature space was discretized. At each pixel position the maximal activity for a total number of 36 different orientations, reaching from 0° to 175°, was calculated and is shown in Fig. 3(c).

### 3.2 Finding good Features

The method used to display the quality of an "intended" completion in Fig. 3(c) is not suitable for real world problems. It discretizes the feature space and is computational intractable because of the brute force data generation over the whole space.

We therefore propose to use a sampling technique which has been proven feasible for path planning and protein folding, the transition based rapidly exploring random tree (T-RRT) [4]. T-RRT extends classical RRT to find good, cost-efficient paths in the presence of a cost function defined on the configuration space, e.g. finding paths along a valley in a mountainous region. Thus it applies a transition test supplementary to the state validity checking of RRT. The transition test, as presented in [4], needs a cost function $c$ which evaluates the cost of a newly sampled configuration $q$. The probability of a transition from state $q_t$ to a new state $q_{t+1}$ is then determined by the Boltzmann distribution:

$$p_{t,t+1} = \begin{cases} e^{-\frac{\Delta c_{t,t+1}}{K \cdot T}} & if\ \Delta c_{t,t+1} > 0 \\ 1 & else \end{cases} \tag{7}$$

where $\Delta c_{t,t+1} = \frac{c_{t+1} - c_t}{dist(c_t, c_{t+1})}$, which determines the slope of the cost along the path. $K$ is a constant which normalizes the costs. As proposed in [4], we set $K = \frac{c_{start} + c_{goal}}{2}$. The parameter $T$ is a temperature to allow a simulated annealing behavior. If the transition test is successful, the temperature is decreased by a factor $T = \frac{1}{a} \cdot T$. If the test fails for a given number of steps $nFail$, the temperature is increased by $T = a \cdot T$. Here we also use the same values as in [4], namely $a = 2$ and $nFail = 100$.

### 3.3 Integration

With the given T-RRT algorithm and the quality measurement from (6), it is easily possible to create a cost function:

$$c(v_{new}) = 1 - Q(v_{new}) \tag{8}$$

which prefers paths along high quality features, thus exploring the "intention field" towards well fitting completions. We can consider the known features of the CLM as points on a trajectory through this space. The task of the T-RRT algorithm is to connect these points and fill in the sparse regions by finding good amendments, for example to complete the shape from Fig. 3(c).

## 4 Evaluation

To evaluate the proposed approach, we will first test our technique in an interaction game, where the task for our system is to anticipate which figure a user has in mind and complete it accordingly. The second part of the evaluation uses artificially generated shapes and introduces an error margin to gain a quantitative measure how good the generated features of our system are.

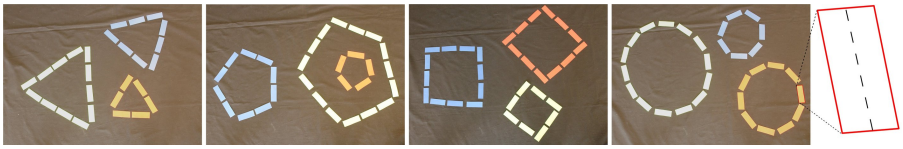


**Fig. 4.** Color labeled training patterns for the completion task. The rightmost part of the image shows the result of the used feature extraction. For each colored block an oriented rectangle is fitted and the longest axis of this rectangle is used as input for the CLM.

### 4.1 Interaction Game

The experimental evaluation consists of a cooperative figure completion task. At first we present some color labeled example figures to our system from which the interaction functions are learned. Each of the four images from Fig. 4 is used to learn the interaction function for a layer class. Each colored rectangle has the size of a wooden block from the Jenga game, as shown in Fig. 4. From these images the features are extracted using a simple color threshold and by applying a box fitting algorithm from *OpenCV*[1] to the filtered images. The longest axis of these boxes is used as an individual input feature for the CLM. The goal of the system is to anticipate which figure the human player has in mind and to complete it accordingly by suggesting a well-fitting location to place the next Jenga block using the presented approach. The actual placement of the wooden blocks was not yet realized by a robotic pick-and-place program, due to minor technical issues. Occasionally the human player may place a block by himself.

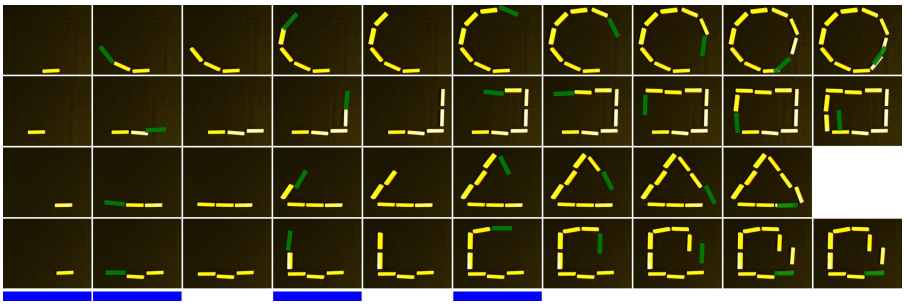
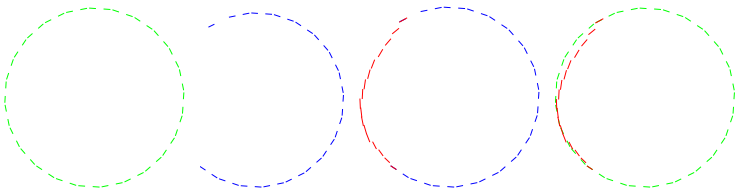


**Fig. 5.** Results from the cooperative shape completion task. Each of the four rows shows a sequence of the task. The color coding at the bottom of the figure indicates if it is the user's or the system's turn. A blue bar represents an action performed by the user and no bar stands for an action executed by the system. The green bars in each frame denote the anticipation of the system for the following frame. After the sixth frame the system is completing the shape autonomously.

[1] Open Source Computer Vision - http://opencv.org



(a) Original (b) Sparse (c) "Intentions" (d) Comparison

**Fig. 6.** Illustration of the generation and evaluation of artificial shapes. The shape in the leftmost image serves as reference, which is used to generate a sparse shape as shown in Fig. 6(b) and as ground truth data for the error calculation. In Fig. 6(c) the completing features generated by our approach are shown in red. The rightmost image compares the completing features from Fig. 6(c) to the ground truth data. The mean mutual distance between these features is used as error margin.

When a new block is placed on the table, our system explores the neural potential in the vicinity of this block towards unoccupied regions. Because the geometry of a Jenga block is known and the task is to place one block at a time, the exploration radius is limited to the length of one block.

Four game sequences are shown in Fig. 5 corresponding to different geometric shapes to be laid out. Each row depicts a single trial which shows the sequence from left to right. The bottommost row indicates when our system or the user performed the shown action in that column. A blue label indicates that the action was performed by the user whilst no label represents that our system decided where to place the next block. Additionally, a green block in the sequence of images shows the anticipation of our system for the following frame.

The procedure was similar for all trials. The first two Jenga blocks were placed by the user. From block three to six the system and the user were taking turns. Eventually, the system should complete the figure autonomously.

The evaluation shows overall good results. Especially the circular shape in the topmost row is rapidly recognized, which can be seen by the first first anticipation of the system in the second frame. The other shapes were recognized in the fourth frame, because the lines in the previous three frames are too ambiguous. After getting a hint in the fourth frame through the user's action, the system detects the rectangular respective triangular shape. Also the autonomous completion after the sixth frame works well in three of four shown trials. Only the rectangle in the last trial is not completed as expected. This may have the cause that the training pattern from Fig. 4 contains a shape with an edge length of two blocks, which corresponds to the anticipation of our system. Although the used learning technique generalizes over different sizes and rotations of the presented shapes, which can also be seen by the figure created in the second sequence, which is not present in the training image, the presented training shapes may have not been variant enough to fully exploit the generalization ability of the learning approach.

### 4.2 Artificial Data

To gain a quantitative error margin for the proposed approach, we evaluate the process with artificially generated shapes. To this end, a shape with varying size and rotation is generated and stored as ground truth data for later comparison. We will denote this complete shape as $C$. From this shape $C$, features are randomly removed to create a sparse shape, hereafter named $S$. Our approach is then used to close these gaps by generating completing features. The set of these features will further be denoted as $H$. The used error margin (9) is the mean mutual distance of generated features from $H$ to their closest corresponding feature in the original set $C$. An exemplary overview of this procedure is shown in Fig. 6. The first image shows the complete shape which is used as reference. In the second image a number of features is removed to create shape $C$. The result of the "intention" completion can be seen in Fig. 6(c). An overlay of the "intended" completions and the original shape is illustrated in Fig. 6(d). From the features of Fig. 6(d) the error is calculated as:

$$E(C,H) = \frac{1}{|H|} \sum_{h \in H} \min_{c \in C} ||d(c,h)|| \tag{9}$$

We generated four different kinds of shapes, namely a triangle, a rectangle, a pentagon and a circle with varying sizes and orientations. From these shapes up to 40% of the feature were removed randomly in steps of 10%. For each shape and percentage of removed feature, 50 shapes are randomly generated. The generated shapes have an average of 2.8 distance between two features. In table 1 the average mutual distance over the 50 trials per shape and removed percentage is shown. There are no significant outliers in the data and given the distance

**Table 1.** Distance between "intended" completions and original shape.

| **Shape** | **percent removed** | | | | **Shape** | **percent removed** | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **10%** | **20%** | **30%** | **40%** | | **10%** | **20%** | **30%** | **40%** |
| **Triangle** | 4.77 | 4.99 | 4.64 | 4.62 | **Rectangle** | 3.39 | 4.71 | 4.83 | 4.70 |
| **Pentagon** | 3.18 | 4.65 | 3.95 | 4.84 | **Circle** | 2.93 | 3.59 | 3.88 | 3.75 |

between the generated features of 2.8, the mutual distance is small enough to say that our approach can create reasonable amendments in the absence of data. Since the mutual distance in case of the circle is overall smaller than the mutual distances of the other shapes, this suggests that our approach slightly prefers smooth continuations to corners.

## 5 Conclusion

We presented an approach which exploits the perceptual grouping capabilities of the Competitive Layer Model and introduces a technique which extends these

grouping capabilities towards the automatic generation of new data. The approach shows reasonable error margins in the evaluation with artificial data, which indicates good completion abilities. It is further evaluated in an interaction scenario which utilizes the generative aspect of this technique to anticipate the intentions of the human partner. Although the scenario is rather simple, the results are convincing that this extension of the CLM capabilities can be useful in a broader domain and more complex scenarios, because the only part which has to be supplied by a user is a compatibility function in the feature domain. This property also eases the adaption of the presented approach in comparison to more specialized data generation techniques like, for example, [2,8]. Given the findings from [10] and the presented compatibility measurement for actions, we strive to extend the CLM based segmentation of actions towards anticipating good completing actions for cooperative human-robot tasks.

## References


1. Awais, M., Henrich, D.: Proactive premature intention estimation for intuitive human-robot collaboration. In: IROS 2012. pp. 4098–4103. IEEE (2012)
2. Eslami, S.A., Heess, N., Winn, J.: The shape boltzmann machine: a strong model of object shape. In: CVPR. pp. 406–413. IEEE (2012)
3. Gehrig, D., Krauthausen, P., Rybok, L., Kuehne, H., Hanebeck, U., Schultz, T., Stiefelhagen, R.: Combined intention, activity, and motion recognition for a humanoid household robot. In: IROS, 2011. pp. 4819–4825. IEEE (2011)
4. Jaillet, L., Cortes, J., Siméon, T.: Sampling-based path planning on configuration-space costmaps. Robotics, IEEE Transactions on 26(4), 635–646 (2010)
5. Kelley, R., Nicolescu, M., Tavakkoli, A., King, C., Bebis, G.: Understanding human intentions via hidden markov models in autonomous mobile robots. In: HRI, 2008. pp. 367–374. IEEE (2008)
6. Köhler, W.: A source book of Gestalt psychology. Kegan Paul, Trench, Trubner & Company (1938)
7. Meier, M., Haschke, R., Ritter, H.: Hallucinating image features to supplement perceptual groups. In: Workshop New Challenges in Neural Computation (2011)
8. Ming, Y., Li, H., He, X.: Connected contours: A new contour completion model that respects the closure effect. In: CVPR. pp. 829–836. IEEE (2012)
9. Nattkemper, T., Wersing, H., Schubert, W., Ritter, H.: A neural network architecture for automatic segmentation of fluorescence micrographs. Neurocomputing 48(1-4), 357–367 (2002)
10. Pardowitz, M., Haschke, R., Steil, J., Ritter, H.: Gestalt-based action segmentation for robot task learning. In: Humanoids 2008. pp. 347–352. IEEE (2008)
11. Ritter, H.: A spatial approach to feature linking. In: Int. Neural Network Conference, Paris (1990)
12. Weng, S.: Data driven learning for feature binding and perceptual grouping with the Competitive Layer Model. Bielefeld University (2005)
13. Weng, S., Wersing, H., Steil, J., Ritter, H.: Learning lateral interactions for feature binding and sensory segmentation from prototypic basis interactions. Neural Networks, IEEE Transactions on 17(4), 843–862 (2006)
14. Wersing, H., Steil, J., Ritter, H.: A competitive-layer model for feature binding and sensory segmentation. Neural Computation 13(2), 357–387 (2001)

# Learning as an essential ingredient for a tour guide robot

Sven Hellbach, Frank Bahrmann, Marc Donner, Marian Himstedt, Mathias Klingner, Johannes Fonfara, Peter Poschmann, Richard Schmidt, Hans-Joachim Boehme*

University of Applied Sciences, Artificial Intelligence Lab, Friedrich-List-Platz 1, 01069 Dresden, Germany

**Abstract.** Even though, that over the last years the methodical fundament for robotics has become more and more stable, the integration of these different methods to a useful practical application offers some challenging aspects. This paper describes the design and development of a tour guide robot already being used in his scenario environment. The paper will show how the different methods are used and how still a research on the different methods itself can be made possible.

**Key words:** museum tour guide robot, navigation, map building, people tracking, body pose estimation, dialog modeling

## 1 Introduction

The idea of a tour guide robot in a museum environment has already been addressed several times in the past ([1–3] only to name a few). Within those projects the museum specific part of the story is usually limited to showing around visitors and present information to the exhibits or in the sense of a concierge robot to point out interesting exhibitions.

For our project of a museum tour guide robot, the interactive and entertaining presentation of information is one of our major goals. During the duration of our project the robot is already present in the museum. Our project partner *Technische Sammlungen Dresden* which is a collection for history of technology allows and encourages us to evaluate our methodical development in an exhibition for vintage computer hardware.

Within the exhibition the robot is meant to wait at the entrance and welcome arriving visitors. For starting a verbal dialog with the visitors the robot starts to introduce itself as well as the entire exhibition. This first dialog between the visitor and our robotic system is supposed to collect information about the user's intention, expertise and interest. On basis of this information the tour through the exhibition will be tailored individually. For example the number and order of the exhibits during the tour can change depending on the estimated user's preferences.

At each of exhibits on the tour the robot will be able to decide, which information about the exhibit is to be presented. This is inferred from previous dialog situation, in which the user has shown interest in particular categories, like history facts or technical details. Furthermore, if the interest of a single user is drawn to another exhibit the robot should be able to change the tour accordingly.

With that plot in mind, bringing an assistance robot into our all day living, the system needs to be adaptive and hence has to be able to make intelligent decisions. Challenging demands of such application go far beyond the abilities of simple state machine like algorithm. In particular, our robot companion needs understand its environment and - even more important - the persons within the environment.

As already stated, our project demonstrator is situated in a historical computer science exhibition. From our experience, a lot of users are not particularly interested in vintage computer hardware. Furthermore, one has to argue for such projects about the benefits over a human presenter.

* This work was supported by ESF grand number 100076162

(a) (b)

**Fig. 1.** Two possible scenarios of augmented reality with our robot: (a) Robot projecting to a plain wall, (b) Robot projecting onto a CRT display, 'reviving' the old hardware.

For both problems we came up with the idea of mounting a video projector on our robot. The benefit of that idea is at least two-fold: (1) We are able to project the content about the exhibits e. g. on a wall instead of only presenting it on a robot mounted screen. Hence, the information can be made available to a larger audience (see Fig. 1a). (2) It is furthermore possible, to project details directly on the exhibits, e.g. to highlight certain parts for detailed descriptions .

An especially appealing scenario is the revival and live demonstration of old computers by running a simulator on the robot and projecting the simulated screen output onto a CRT display. This way we can demonstrate the full capabilities of several old computers without having to activate the historic hardware itself (see Fig. 1b).

To fulfill the described story several aspects need to be considered which can be roughly divided into navigation, described in Section 2, and human-robot-interaction (HRI), discussed in Section 3. For the navigational part a map of the environment is needed in which the robot needs to find its current position as well as for planing the path to the target position. This paper discusses our framework for building such a map as well as some ideas on how such a map, which is large for our environment, can be handled efficiently.

Furthermore, for being able to interact with the visitors the tracking of persons plays an important role. A tracking of the persons position, his or her face and even the entire body pose takes place. The different kinds of trackers allow to reason about the persons intention by understanding their motion. These estimations can than be used for dialogue modeling to be able to conveniently interact with people.

Finally, we have to face the problem, that the system is evaluated while already operating in a public environment. Having only a partially functional system would not be accepted by the visitor making tests and evaluation difficult or even impossible. Hence, we came up with the idea to gap the missing or not yet functional parts of our system by a human operator in a Wizard-of-Oz like manner as presented in Section 4.

## 2 Navigation

To be able find its position the robot needs a way to perceive its environment and compare these sensor readings with a model of the environment. For our approach an occupancy grid map based on the data coming from a laser scanner is applied for localization and path planing. The map can be enriched with semantic information, allowing to attach information needed e.g. by the dialog system. Since the laser scanner is limited to perceive objects in a plane parallel to the floor at about knee height, an additional 3d obstacle avoidance system enables to detect objects, like e.g. tables.

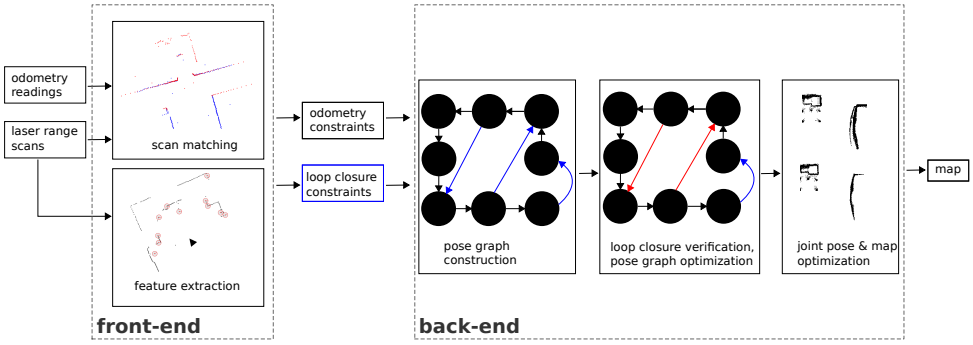



**Fig. 2.** Overview of the navigation framework. The front-end has the task to provide spatial relations of the robot poses, while the back-end maintains these poses in a graph structure for estimating the traveled path of the robot.

### 2.1 Map Building

Providing the path of the area traversed is known, a map can easily be built. For most of the cases these paths are not available respectively hard to obtain at high accuracy for indoor environments. If both, map and path, are unknown, the robot has to concurrently maintain estimates about its position as well as the traversable environment which is well known as the Simultaneous Localization and Mapping (SLAM) problem.

We use a SLAM framework [4] enabling to build highly accurate maps using laser range finders which is illustrated by Figure 2. This includes a front- end providing spatial relations of robot poses by means of local motions and loop closures. This information is passed to a back-end that maintains pose relations and estimates the path travelled by the robot. A joint pose and map optimization is carried out afterwards given the estimated trajectory.

The front-end provides an initial estimate using laser scan matching about the successive robot poses. In addition to that the front-end is responsible for recognizing previously observed places and thus passes loop closure candidates to the back-end. This is done by extracting FLIRT[1] features from laser range scans which are matched to those obtained from the previous scans. The most important requirement of a SLAM system is its ability to detect and incorporate information about places that have already been visited which are referred to as loop closures. Again, we rely on FLIRT features with an additional RANSAC-based consistency check.

Thanks to the front-end we are given an initial graph of robot poses and relations expressing spatial constraints between those. The graph consists of the different poses on the trajectory as vertices and the actions of getting from one pose to another as edges. The optimization of this graph to handle false loop closure detections is the task of the back-end. The final map is built using sparse surface adjustment (SSA).

For the environment the maps usually become large depending on the desired resolution. In [5, 6] an Non-negative matrix factorization (NMF) based approach has been presented allowing a sparse coding of the occupancy map. This is possible since NMF extract a set of basis primitives which contain the information about the characteristics of the environment, like corner, horizontal wall, etc. Since these basis primitives are generated based on the available map, they are specific for the current environment. Furthermore, based on the sparse coded maps a histogram like representation is extracted. This kind of representation can be used to optimize the localization.

### 2.2 Semantic Labeling

Putting the robotic system into service involves building a map of the new environment including the labeling of the parts of the exhibition. Even while the robot is already in service adding or

[1] Fast Laser Interest Region Transform

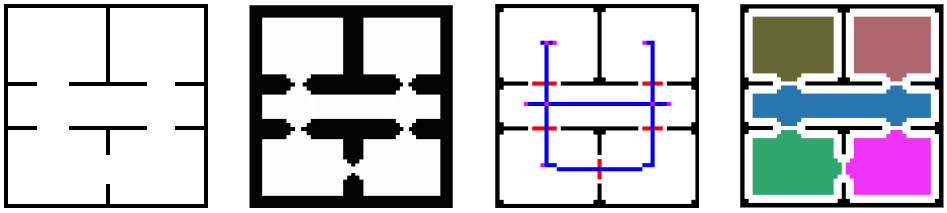



**Fig. 3.** (a) The raw map (where white pixels are free and black pixels are occupied) and subsequent processing steps: (b) The dilated and gap closed binary map. (c) The map skeleton (blue) with starting and intersection points (purple) and critical lines (red). (d) The constructed areas.

moving exhibits is an expected task. For the chosen scenery the staff of the museum should be able to do this. The method described in detail in [7] allows the museum's staff to "teach" the robot about new constellations.

In our opinion the best way to accomplish this familiarization is by interacting with the robot in a very humanlike manor. This idea is inspired by the human habit to show around new co-workers. To handle this scenario it is necessary for the robot to detect and follow humans, to interact with them through a dialogue and to recognize rooms by splitting a built map in semantic parts - a topological map.

Our method is divided in three subsequent steps (see figure 3b-d). To explain the single stages, a rather simple simulated occupancy grid map is used as raw material (see Fig. 3(a)). The first step is filtering and dilating the original map (Fig. 3(b)), followed by thinning (blue line in Fig. 3(c)) and the last step is separating areas (Fig. 3(c) and (d)).

### 2.3 Local Navigation

We use a navigation system using laser based localization supported by a 3D obstacle segmentation system [8]. Fox et al. [9] introduced the Dynamic Window Approach (DWA) for collision avoidance, that became very popular over the last decade. We utilize a version of DWA with Lyapunov stability criterion [10, 11], a widely used method in control theory. The goal of the DWA is to reach a waypoint that is located on the planned path in a defined distance to the robot, so it will try to follow the path approximately, while avoiding obstacles, until the target is reached.

### 2.4 3D Obstacle Avoidance

Using only a laser range finder for navigation would not allow to detect each and every obstacle. In particular, exhibits placed on tables or within glass vitrines are completely ignored. That's why, we enrich the laser detection with information coming from a time of flight camera mounted on the head of the robot [12].

After pre-precessing the depth data to reduce noise, a floor plane is estimated dynamically for each frame. We select a set of points that were classified as free space in the previous depth image to get an estimate for the ground plane within the actual depth image. To reach a high efficiency of the frame-wise ground plane correction, we refine the suspected ground plane position with linear regression over some selected range measurements. After correcting the base plane position, obstacles are separated from the base plane by using a fixed and an amplitude based threshold.

## 3 Human-Robot-Interaction

For providing the possibility of an interaction between our robot and the human visitor a number of components are necessary. First of all it is import to recognize persons at all and furthermore to

know their position around the robot. To ensure this, a people tracker is essential. For involving the surrounding visitors into an interaction a spoken dialog is necessary. To make the dialogue adaptive to the needs of the visitors a dialog model needs to collect all available subtile informations from the user, e.g. the body language. People tend to turn away from their dialog partner as soon as the interest in the dialog fades away. There are even studies that show, that it is possible to infer the mood from the way a person moves. To perceive both information a human motion estimation system is developed [13]. The presentation of information to a larger audience or even augmenting the exhibits using a video projector relies on a system for projection correction.

### 3.1 People Tracker

In order to look at people, the robot needs to know the position of the head of each person (and the position of its own head, as we operate within a global coordinate system). The measurements of all sensors will be transformed into the same global coordinate system. Most of the sensors are not able to estimate the position in all three dimensions – laser and sonar sensors can only detect the legs of people and do not have the possibility to sense the elevation of the head, while cameras can only give rough estimations of the distance to the detected head based on assumptions of the size of that head. Therefore, the information of different sensor cues should be fused and uncertainties should be accounted for.

We use a tracking-by-detection approach, where a sensor cue detects people at certain moments and passes those detections to the tracker, which updates tracks to combine them with previous detections [14]. To make tracking more robust and being able to model uncertainties, we use a Kalman filter to estimate the positions of each person independently of others. To keep the sensor cues consistent and make integration of new cues easy, their detections will be converted into normal distributions describing the estimated position of a detected persons head in three-dimensional Euclidean space before passing them to the tracker.

As sensor cues we use leg hypotheses being detected in a laser range finder, a face and head-shoulder-contour hypotheses both based on the Viola and Jones algorithm detected in an omnidirectional gray-level as well as a directed infrared image.

Because sensor measurements are uncertain and prone to errors, probabilistic approaches are most often used in mobile robotics for state estimation. Instead of including the positions of all people into one joint state space, we track them independently of each other so the state space has a low dimensionality and the problem remains computationally efficient. We assume the state of a person to be normally distributed and use a Kalman filter for estimation.

### 3.2 Dialog System - User model

Since a major demand of the defined task is the interaction with the user, the dialog system is a central part of the robot. A dialog policy controls the next chosen speech utterance. This policy can either be designed by an expert or training using reenforcement learning strategies. The rewards for the reenforcement learning can be derived from the user behavior (e.g. a negative reward for an aborted interaction) or by a human operator observing the ongoing dialog.

The decision of the policy is based on a knowledge base. Starting with the world model the robot knows its position from which the likelihood of request about certain local exhibits can be estimated. The domain model simply provides facts of the exhibits and other answers to possible question. The user model collects information about the users interest or current mood. Together with the history of previous user behavior the most likely intention of the user can be inferred. On top of that the task model gives the dialog a structure, simply speaking a greeting should take place before presenting the exhibits followed by a farewell.

### 3.3 Human Motion Estimation

The body pose is estimated with an algorithm [15], which enhances the approach presented in [16]. It relies on the image of a depth camera from which a Self-Organizing Map (SOM) is extracted

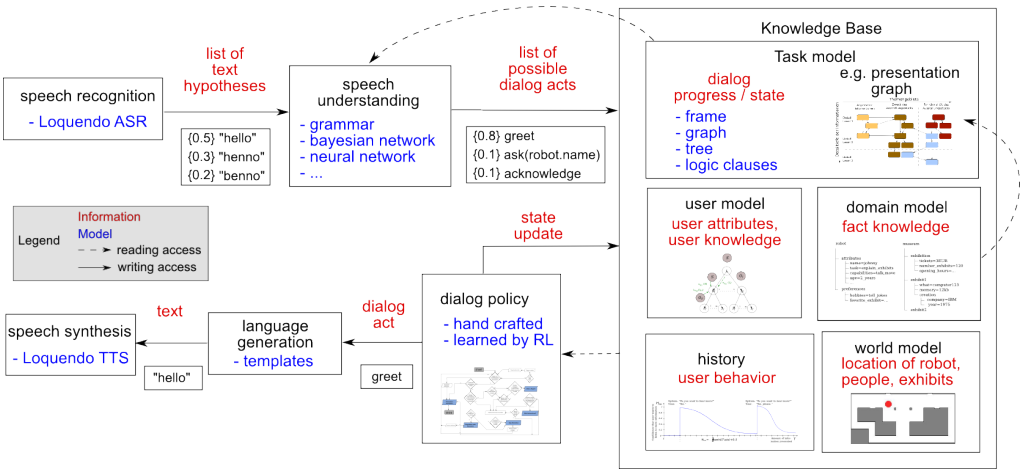



**Fig. 4.** Overview of the dialog system. The dialog policy is meant to chose the best fitting response to the behavior of the user. The decision is based on a knowledge base consisting of a task model, a user model, a domain model, a history and a world model. The knowledge is a mixture of previously learned information, predefined expert knowledge, and information collected directly from the user. The latter is mainly based on speech input.

to model the human upper body. Crucial in that context is the correct assignment of the SOM neurons to a specific region of the tracked person's upper body. In [16] the best-matching neuron (BMN) for a presented stimulus is determined based on the Euclidean distance with the three spatial dimensions $x, y$ and $z$. Computation of the minimal Euclidean distance seems to be the most straight forward solution. However, it turned out to be insufficient for structured three-dimensional objects like the upper body, irrespective of the applied learning paradigm. Furthermore, depending on the specific technology used, additional problems may appear due to various sources of error and noise. This leads to the situation that sometimes neurons migrate from one part of the upper body to another. Without a verification of the SOM a future subsequent classification of the pose will produce incorrect results and maybe lead to wrong interpretation of the actual situation. Therefore we extended the approach in [16] by reshaping the trained SOM to a skeleton model to estimate the anatomical correctness of the pose. Having generated the skeleton model, incorrect Self-Organizing Maps will be rejected if the subsequent verification failed.

The goal of our approach is the generation of a problem specific parameter vector for a group of neurons. Therefor, the weight vector of a SOM neuron is extended compared to [16] including features like RGB color values, texture and neighborhood descriptors and the three spatial dimensions. By determining the relevant dimensions, a parameter vector or matrix is computed which includes a weight for each dimension of the weight vector of a group of neuron using GRLVQ or GMLVQ respectively [17, 18]. The benefit of this method is binding neurons to a specific region by considering additional parameters in the input vector - rather than only their coordinates.

### 3.4 Projection Correction

To be able to present the information on the exhibits, three steps need to be taken, from which two are addressed throughout the paper. (1) The robot needs to find the way to the exhibit (2) Arriving at the target position, the robot needs to localize itself 'well' enough to have the exhibit in the possible field of projection. (3) Finally, the distorted projection needs to be straightened to fit the desired part of the exhibit.

For the first two problems the algorithms described in Section 2 are applied. With that, we can localize our robot with satisfying precision so that we can assume our projection target is within

the area coverable by the projector as well as the camera's field of view. Additionally, we restrict ourself to just planar projection surfaces.

The final goal of the projection correction is to find a transformation, that when applied to the projector image, results in a projection which fits a designated projection target. Our first step is to identify the projection target in the camera image. To achieve this, a marker detection system could be utilized. Our future goal is to use 3D object recognition instead. For now it is sufficient to know the four outer corners of the projection target, since we assume the target to be planar and aim for projecting into a rectangular area on the projection surface computing and applying a homography transformation [19].

## 4 System Evaluation Strategy – Wizard-of-Oz

A complex system like a tour guide robot needs a thorough evaluation which has to consider multiple aspects. User acceptance has to be evaluated as well as how well different methods work within the given environment. For both the presence of unbiased visitors is necessary. Hence, the system has to be fully functional all the time, even if some software components are not working reliable enough or are not yet finished.

Furthermore, we believe that having a spoken dialog with museum visitors is an important aspect of a tour guide robot. It attracts people and allows to demonstrate an intelligent system. One reason why todays guide robots still lack complex dialog capabilities is, that speech recognition is a major unresolved problem. Developing such a dialog system under real world conditions is a challenging task. An unfinished system would leave the visitors unsatisfied or even frustrated. Hence, we have decided to create the illusion for the visitor, that the robot is already able to talk to them in a natural way. This is achieved by replacing the speech recognition system by a human operator using a Wizard of Oz method [8].

For Wizard of Oz inspired experiments a hidden operator controls the robots dialog system. In order to allow the operator to still react to the visitors, the images from the omni-directional camera as well as an audio stream were transferred to the operators laptop. To select the available phrases, different milestones of the dialog were defined. For all of those milestones answers or reactions to possible questions or situations were assembled beforehand. The intended dialog should consist of the following milestones: (1) greeting, (2) small talk, (3) introduction and explanation of the robot, (4) suggestion of a tour through the exhibition, and (5) explanation of exhibits.

The operator can select the robots target position and initiate the driving mode, which is completely autonomously. While the robot navigates, his head is facing in the direction of movement. This allows persons walking towards the robot to guess the robots intention to pass. Such a behavior is one of the many subtle steps towards a socially acceptable navigation. During the dialogs, the head of the robot is facing the dialog partners.

## 5 Conclusion

The proposed methods and the way of combining and evaluating them is not limited to the museum tour guide scenario. Most non-technical scenarios have to deal with the combination of navigation and human-robot-interaction. Other possible applications can be found in a numerous way in the literature. One of the most frequently discussed are elderly care scenarios, for which the robot is meant to facilitate the routine work of nursing and security personal, as well as a help and support for the retirees.

The paper describes an ongoing project. Hence, some of the described sub-systems are still in prototypical state. The evaluation of these systems has already been published as it was mentioned throughout the paper. However, an evaluation of the entire system is an open task.

Even though, the described project already offers an impressive gain for the visitors of the exhibition, the number of problems and the possibilities to extend the scenario are still numerous. The dialog system currently relies on text explicitly written by the curator of the exhibition.

However, for example IBM's Watson project shows that it is possible to integrate larger databases offering more detailed information for the visitor.

Furthermore, a possible use of the museum tour guide might be to make it accessible online. In that way, a visit over night or on less crowded days would be available from at home.

Since the goal is to be as user adaptive as possible, it is fundamental to get as much information from the user as possible, without asking. Beside motion interpretation a "large" cue for estimating the users' mood is its prosody. Since we already do audio processing for speech recognition, this will be the next step.

## References


1. Burgard, W., Cremers, A., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., Thrun, S.: Experiences with an interactive museum tour-guide robot. ArtificialIntelligence **114**(1-2) (2000)
2. FROG Consortium: Deliverable: D1.1, functional requirements, interaction and constraints. Technical report, Universiteit van Amsterdam (UvA), Ydreams - Informatica S.A. (YD), Idmind - Engenharia de Sistemas Lda (IDM), Universidad Pablo de Olavide (UPO), Imperial College of Science, Technology and Medicine (ICL) (2011)
3. Thrun, S., Bennewitz, M., Burgard, W., Cremers, A.B., Dellaert, F., Fox, D., Hähnel, D., Rosenberg, C., Roy, N., Schulte, J., Schulz, D.: Minerva: A second-generation museum tour-guide robot. In: In Proceedings of IEEE International Conference on Robotics and Automation (ICRA 1999). (1999)
4. Himstedt, M., Keil, S., Hellbach, S., Böhme, H.J.: A robust graph-based framework for building precise maps from laser range scans. In: ICRA. (2013)
5. Hellbach, S., Himstedt, M., Böhme, H.J.: Towards Non-negative Matrix Factorization based Localization. In: ECMR. (2013)
6. Himstedt, M., Hellbach, S., Böhme, H.J.: Feature extraction from Occupancy Grid Maps using Non-negative Matrix Factorization. In: NC2 2012, Graz (AT), Machine Learning Reports (2012) 97–100
7. Bahrmann, F., Hellbach, S., Böhme, H.J.: Please tell me where I am: A fundament for a semantic labeling approach. In Wlfl, S., ed.: Poster and Demo Track of KI. (2012) 120–124
8. Poschmann, P., Donner, M., Bahrmann, F., Rudolph, M., Fonfara, J., Hellbach, S., Böhme, H.J.: Wizard of Oz revisited: Researching on a tour guide robot while being faced with the public. In: RO-MAN 2012. (2012) 701 – 706
9. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. Robotics Automation Magazine, IEEE **4**(1) (mar 1997) 23 –33
10. Ogren, P., Leonard, N.: A convergent dynamic window approach to obstacle avoidance. Robotics, IEEE Transactions on **21**(2) (april 2005) 188 – 195
11. Berti, H., Sappa, A., Agamennoni, O.: Improved dynamic window approach by using lyapunov stability criteria. Latin American Applied Research **38(4)** (2008) 289–298
12. Donner, M., Poschmann, P., Klingner, M., Bahrmann, F., Hellbach, S., Böhme, H.J.: Obstacle detection for robust local navigation through dynamic real-world environments. In: Workshop on Color-Depth Camera Fusion in Robotics at IROS. (2012)
13. Pollick, F.: The features people use to recognize human movement style. In: Gesture-Based Communication in Human-Computer Interaction. Volume 2915 of LNCS. Springer (2004) 10–19
14. Poschmann, P., Hellbach, S., Böhme, H.J.: Multi-modal People Tracking for an Awareness Behavior of an Interactive Tour-Guide Robot. In: ICIRA. Volume 7507 of LNCS. Springer Berlin / Heidelberg (2012) 666–675 10.1007/978-3-642-33515-065.
15. Klingner, M., Hellbach, S., Kästner, M., Villmann, T., Böhme, H.J.: Modeling Human Movements with Self-Organizing Maps using Adaptive Metrics. In: NC2 2012, Graz (AT), Machine Learning Reports (2012) 14–19
16. Haker, M., Böhme, M., Martinetz, T., Barth, E.: Self-Organizing Maps for Pose Estimation with a Time-of-Flight Camera. In: Proceedings of the DAGM 2009 Workshop on Dynamic 3D Imaging. Volume 5742 of Dyn3D '09., Berlin, Heidelberg (2009) 142–153
17. Hammer, B., Villmann, T.: Estimating Relevant Input Dimensions for Self-organizing Algorithms. In: Advances in Self-Organizing Maps, Springer (2001) 173–180
18. Schneider, P., Biehl, M., Hammer, B.: Adaptive relevance matrices in learning vector quantization. Neural Comput. **21**(12) (December 2009) 3532–3561
19. Donner, M., Himstedt, M., Hellbach, S., Böhme, H.J.: Awakening history: Preparing a museum tour guide robot for augmenting exhibits. In: ECMR. (2013)

# Learning Dialog Management for a Tour Guide Robot Using Museum Visitor Simulation

Johannes Fonfara, Sven Hellbach, and Hans-J. Böhme

Artificial Intelligence Lab, University of Applied Sciences, Dresden, Germany
{ fonfara; hellbach; boehme }@htw-dresden.de

**Abstract.** In this paper we address the problem of dialog system modeling for a tour guide robot whose task is to present exhibits to museum visitors. We present an approach for multimodal, dialog-directed interaction using a Markov decision process. A user simulator built upon our experience and data from Wizard-of-Oz experiments is used to train the dialog manager. Experimental results confirm the suitability of our approach for the application and motivate its use in the real world setting.

**Keywords:** Tour Guide Robot, Dialog System, Reinforcement Learning

## 1 Introduction

In this paper we present ongoing work on a dialog system with application to an autonomous tour guide robot. Motivated by frustrating and non-intuitive user interfaces, our research goal is to improve human-machine communication by methods that infer user goals (i.e. what users want) in function of their behavior (i.e. user actions the machine can observe). The task in our scenario is as follows: In a museum, our service robot looks for visitors to whom it can offer a museum tour. Upon contacting persons and a short introduction, it will take them from exhibit to exhibit, present information, and answer their questions. Crucial to its success is to know how much and which information to present about each exhibit, based on the actively estimated interest of the audience.

**Fig. 1.** The service robot interacts with visitors at the museum site.

In real-world dialogs there is a vast number of distinct situations. With this in mind, it makes sense to build a system that can learn its policy (a mapping from situations to actions), avoiding laborious hand-crafting of the dialog manager behavior. Hence, reinforcement learning (RL) has evolved within the past years as state of the art for optimizing dialog systems, which gives various advantages:

Multimodal observations are fused into an appropriate state representation, which, by statistical knowledge, allows for probabilistic reasoning and inference of user goals. Furthermore, RL frameworks can learn from experience, possibly improving their policy the more they interact with their users.

Implementing a tour guide robot that learns from interactions with its users has, to the best of our knowledge, not been addressed yet by the machine learning community. Comparable guide robots possess either just basic dialog capabilities [1] or focus on small talk [2]. Recent development has shown a growing interest for optimizing dialogs in terms of user satisfaction [6], which is also in our interest.

## 2 Reinforcement Learning for Dialog Systems

Markov decision processes (MDPs) were used for spoken dialog systems since more than a decade [5]. Formally, an MDP is a tuple $M = \{S, A, T, R\}$ where $S$ is the set of states, $A$ is the set of system actions, $T(s'|s, a)$ is a set of state transition probabilities, and $R : \mathcal{S} \mapsto \mathbb{R}$ is a reward function which maps states to real-valued rewards. The MDP model for dialogs operates as follows: At each time step the world is in some current state $s \in S$ which is assumed to be known exactly. Subsequently the machine selects an action $a \in A$ and applies it. The user's reaction is observed thereafter, leading to a transition of the current state $s$ to a successor state $s'$, where $s'$ depends only on $s$ and $a$. For this transition the machine receives a real-valued reward which is defined by $R$.

For real-word applications the state of the world can not be known exactly. Sensor noise and recognition errors result in faulty state assumptions and poor decision making. A model that fits better to the real wold is the partially observable Markov decision process (POMDP), an extension to the MDP, which maintains a probability distribution over states. Exact computation of this belief state is intractable for most applications with more than a small number of states, but approximation algorithms were developed, promising a tractable solution for our dialog application [9].

In this early stage of development we preferred to use an MDP rather than a POMDP model since a variety of complexity problems do not have to be addressed yet. This work aims primarily to investigate the usability of RL-methods for our tour guide application and we chose the simpler model in order to save time. However, we plan to cast our system as a POMDP in a future version, and most of the model can be adopted without alteration.

## 3 Simulation of Museum Visitors

The transition function $T(s'|s, a)$ in an MDP can be very complex, depending on the state representation and size. Fortunately, we do not have to specify it when using a learning algorithm that uses random sampling. To generate those samples we built a simulated user for the system to interact with. This technique has been applied soon after RL was first proposed to optimize dialog systems,

**Table 1.** Behavior defining attributes of the simulated user. All attributes are probability distributions gathered from evaluated Wizard-of-Oz experiments.

| | |
|---|---|
| • # exhibits wanted | • action when frustrated (goodbye/leave) |
| • # information wanted per exhibit | • likelihood of leaving when bored |
| • # mistakes tolerated before quitting | • response on no question (negate/silence) |
| • # consecutive questions tolerated | • response on question (ask/affirm/silence) |
| • response to goodbye (goodbye/silence) | • likelihood of asking questions |
| • likelihood of attention when interested | • likelihood of attention when bored |
| • reaction to correct question answer (feedback/ask next question/silence) | • reaction to apology (feedback/ask same/ask next/silence) |
| • response to the offer of change exhibit (implicit/explicit/silence) | • response to the offer of more information (implicit/explicit/silence) |
| • action when system talks too much (request stop/request change/silence) | • action when system talks too little (request more/silence) |
| • new question likelihood | |

because RL algorithms usually require a large number of training data, too much to gather effectively from Wizard-of-Oz experiments [8].

State-of-the-art user simulators ensure consistent behavior by fixing a goal before the dialog starts, which they try to achieve in the dialog thereafter. However in our scenario, users neither agree, nor stick to one single goal before the interaction; it is rather influenced during the dialog by the system's suggestions or mistakes, as well as by user's preferences and characteristics. The challenge is to find a model able to explain the behavior of users in the given scenario.

We attempt to achieve this as follows: Before every dialog, the character (child/adult) of the user is sampled with probabilities taken from Wizard-of-Oz data. Depending on that, a number of behavior-determining attributes are then fixed as probability distributions for this user (listed in Table 1). The actual behavior is sampled from these distributions when an applicable situation occurs. Furthermore, the user maintains a short-term-memory: Once a response is given, it is memorized and repeated, in case the system asks the same question again.

For every exhibit the user is willing to visit, the number of desired information is sampled before explanations start. This number is decreased by one in every turn the system presents one information unit (e. g. a sentence) about it. If it falls below zero, the user changes to a "bored" state and will now behave according to other attributes, e. g. ask to change the exhibit, act distracted, or leave.

Since MDPs operate on a turn-taking manner, the system expects a user utterance after every action. This is not always the case in the real world, and particularly when the system utters long explanations of exhibits, it can not always perceive what visitors say due to its self-emitted noise. Therefore, the simulator generates a `silence` action with high probability when the system action is `present`. This forces the learning algorithm to pay more attention to nonverbal cues, in order the anticipate interest during its presentations.

In our experiments visitors occasionally asked questions to the robot. We do not simulate actual questions being asked yet, only the fact that they occur, and introduce simulated recognition errors. When a question was correctly recognized, an answer can be given to the user, otherwise the system may ask to repeat or apologize for the error.

Additionally, there is a priority mechanism which aims to explain why users not always give an expected answer: For instance, if the robot asked: "Do you like to hear more?", but the user has an urgent question on his mind, he might ask it instead of answering the earlier prompt.

Much attention was paid to simulate some sort of frustration of users when the dialog manager makes mistakes. Since we want to achieve user satisfaction this is essential to the success of our application. Thus, every time the user asks or requests something and does not immediately get the expected response, a frustration counter is increased. If a tolerance threshold is overstepped, the simulated user quits the dialog by leaving.

## 4 An MDP Representation of the Dialog System

In order to cast our dialog manager as an MDP, we need to define the state space, the action set, and the reward function. We discuss them in the following.

Firstly, the state space $S$ could be represented simply by real or natural numbers. Since we want to describe dialog situations, it is much better interpretable to use a set of discrete variables, where a particular assignment of which represents one state of the dialog manager.

For the museum guide application we use the variables listed in Table 2. They divide into two main groups: user *action* and *goal* variables. Action variables are given by the output of our user simulator and are updated during each turn of the dialog, while goal variables are assigned through deterministic rules, based on the previous state and user actions.

In particular, *goals* express the current user desire to hear `more` about the current exhibit, to `change` to another exhibit, or the state of question answering (`QA`). Applied to the robotic platform, user *actions* are perceived as follows: `Attention` is regarded as the degree to which a person is focused on what the machine says, and can be estimated e. g. by processing image data. Similarly, the `movement` of a person is determined by trajectories provided by a people tracker. The user `speech` action is obtained by classifying the output of a speech recognizer using the nine categories as in Table 2.

Secondly, the dialog manager can choose from one of these eight actions: 1) `present` utters the next sentence about the current exhibit 2) `offer_more` asks the visitors if they like to hear more information 3) `offer_change` offers to move on to another exhibit 4) `offer_questions` offers the visitors to ask questions about the current exhibit 5) `change_exhibit` moves the robot to the next exhibit 6) `answer_question` presents an answer to a question 7) `apologize` states an apology when a question was not understood 8) `goodbye` finishes the dialog.

**Table 2.** State variables of the tour guide dialog manager. There are variables to capture the user's last action (attention, movement, speech) and variables to describe the current user goal status (more, change, QA). Additional history variables (end flag, action counters) exist, but are not listed separately .

| Variable | Possible Values | Meaning |
|---|---|---|
| `attention` | attentive, distracted | visually perceived interest |
| `movement` | approaching, interacting, leaving | visually perceived movement |
| `speech` | affirm, negate | response to a yes/no question |
| | requestChange | request to go to next exhibit |
| | requestStop | request to stop explaining |
| | requestMore | request to explain more |
| | askQuestion | question about an exhibit |
| | feedback | positive or negative feedback |
| | goodbye | quit dialog |
| | silence | nothing said |
| `more` | nothingSaid, requestedStop, requestedMore, affirmed, rejected, notResponded | user goal w.r.t. the desire to hear more information |
| `change` | nothingSaid, requestedChange, affirmed, rejected, notResponded | user goal w.r.t. the desire to change the exhibit |
| `QA` | nothingSaid, affirmed, noQuestion, understood, notUnderstood, answered, apologized, feedbackReceived, notResponded | state of question answering |

Lastly, the reward function $R$ compactly describes the desired dialog system behavior. We want to keep it as simple as possible and therefore use the following rules. In every turn, the default reward is zero. When the agent presents information about an exhibit, a small reward (+1) is received, expressing our objective of providing preferably long museum tours. This also makes dialogs of different length comparable. At the end of every dialog, a large reward (+20) is received when the user did not leave by then and all the information he desired was presented. If the user prematurely quits the dialog (caused by frustration), a large penalty (-20) is given to the agent. Any other time the dialog is prematurely quit by the system, it receives a medium large penalty (-5).

## 5 Experimental Results

When using a factored state representation, the number of states grows exponentially with the number of state variables. In our prototype we have 7290 distinct states given by only the state variables discussed above. A common solution to this is to approximate the state space by a vector of features $\phi : S \mapsto [0,1]^k$, where each state is mapped to $k$ binary features. This dramatically reduces the state space and allows a compact policy representation, where only a feature weight vector $\boldsymbol{\theta}_a$ of length $k$ must be learned for every action $a \in A$.

**Table 3.** Basis functions of the feature vector $\phi$, used for state approximation. Each basis function computes a binary value from the state, as listed below.

| | |
|---|---|
| $\phi_1$ = `more` == nothingSaid | $\phi_{12}$ = `QA` == rejected |
| $\phi_2$ = `more` == affirmed \| requestedMore | $\phi_{13}$ = `QA` == understood |
| $\phi_3$ = `more` == rejected \| requestedStop | $\phi_{14}$ = `QA` == notUnderstood |
| $\phi_4$ = `more` == notResponded | $\phi_{15}$ = `QA` == answered |
| $\phi_5$ = `change` == nothingSaid | $\phi_{16}$ = `QA` == apologized |
| $\phi_6$ = `change` == affirmed \| requestedChange | $\phi_{17}$ = `QA` == feedbackReceived |
| $\phi_7$ = `change` == rejected | $\phi_{18}$ = `QA` == notResponded |
| $\phi_8$ = `change` == notResponded | $\phi_{19}$ = #consec. distracted turns == 0 |
| $\phi_9$ = `QA` == nothingSaid | $\phi_{20}$ = #consec. distracted turns == 1 |
| $\phi_{10}$ = `QA` == affirmed | $\phi_{21}$ = #consec. distracted turns == 2 |
| $\phi_{11}$ = `QA` == noQuestion | $\phi_{22}$ = #consec. distracted turns > 2 |

With this approximation, the state-action value function $Q(a, s)$ and optimal policy $\pi^*(s)$ of our MDP problem can be written as follows:

$$Q(a, s) \approx \boldsymbol{\theta}_a \cdot \phi(s) \tag{1}$$

$$\pi^*(s) = \arg\max_a \boldsymbol{\theta}_a \cdot \phi(s) \tag{2}$$

Table 3 lists the basis functions which have shown to be sufficient for our prototype dialog system. Subsequently, several learning algorithms were applied to train the dialog manager. First results with Natural Actor Critic (NAC) [7] and Least Squares Policy Iteration (LSPI) [3], commonly used algorithms for dialog systems, failed to learn a satisfying policy. They often got stuck applying only one or two different actions regardless of the state.

The following is an attempt to explain this behavior: Both algorithms iteratively compute statistics over large batches of sampled dialogs (NAC computes a policy gradient for each weight in $\boldsymbol{\theta}$, LSPI computes the average, per-feature state-action value function). If no exploitation is performed or dialogs are sampled randomly, every dialog is prematurely quit by the simulated user, yielding a large penalty every time. This happens inevitably until a policy is found which does not frustrate the user. However, the sequence of actions necessary to avoid user frustration is very complex and hard to learn when all parameters are updated at once (NAC and LSPI update the parameters $\boldsymbol{\theta}_a$ after every batch).

A solution to this seems to be a gradual update of only some of the the parameters after a single-step reward is received, as opposed to computing statistics over large sets of dialogs off-line. Temporal Difference algorithms such as parameterized Q-Learning and SARSA [4] follow this intuition. We used a discount factor $\gamma$ of 0.9, an eligibility trace decay $\lambda$ of 0.85, and a learning rate $\alpha$ of 0.1 and applied these two algorithms to our problem. Figure 2 illustrates the mean reward gathered during training of the dialog manager against the user simulator. It can be seen that, as the exploration rate drops to zero, the dialog manager gradually makes fewer mistakes and achieves more successful dialogs.

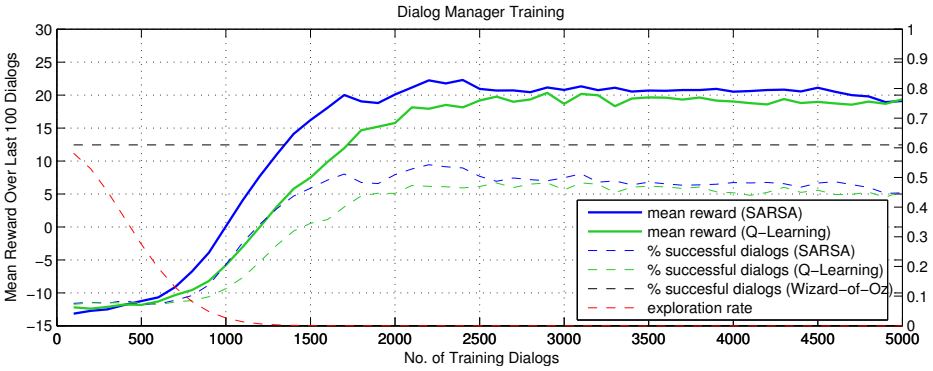



**Fig. 2.** The chart graphs the mean reward over the last 100 sampled dialogs, averaged from 50 training iterations using SARSA and Q-Learning. Apparently, policies trained by the SARSA algorithm gather slightly more reward than those trained with Q-Learning, and both algorithms converge after about 2500 training dialogs.

Caused by hardly predictable user behavior there is a fair amount of failed (prematurely quit) dialogs in the Wizard-of-Oz corpus. We intend to replicate this observation by introducing the same degree of randomness into the simulator while hoping to find a policy that maximizes the rate of successful dialogs. As shown in Figure 2, the success rates of the trained dialog managers yield about 45 - 50%, which does not yet exceed the success rate of our Wizard-of-Oz dialogs (61%) and suggests further extensions of the dialog manager state representation.

## 6 Conclusion and Future Work

All proposed frameworks are part of ongoing research on a dialog system with application to an autonomous robotic tour guide. Our primary goal was to investigate the utility of reinforcement learning for this dialog management problem. Experiments with a simulated user showed that it is possible to learn the principle course of the interaction between a museum visitor and a tour guide robot. In a future version we plan to extend our dialog manager toward more comprehensive interaction capabilities, e. g. introducing small talk behavior.

This work's main contribution is a user simulator which the dialog manager interacts with during the learning process. Using data from Wizard-of-Oz corpus we built a model attempting to replicate the observed behavior. However, it is still incomplete in many aspects and requires more data. For instance, so far we did not attempt to frustrate users on purpose, which is necessary for a better understanding of why visitors quit the interaction.

The generated dialogs reflect our experiences from actual users and motivate further investigations of reinforcement learning methods for the tour guide scenario. Nonetheless, it remains to be evaluated how well the simulator reproduces real museum visitor behavior, e. g. by comparison of the output of the simulator output with Wizard-of-Oz dialogs. Once work on the simulator is completed we

**Table 4.** Shown is a sample dialog between dialog manager and simulator after training using the SARSA algorithm. Interestingly, the dialog manager learned to offer questions when the user was distracted for two consecutive turns. User movements are always "interacting" in this dialog and are thus not listed separately.

| Actor | Speech action | User Attention | Actor | Speech action | User Attention |
|---|---|---|---|---|---|
| System: | present | | System: | present | |
| User: | silence | attentive | User: | silence | attentive |
| System: | present | | System: | present | |
| User: | silence | attentive | User: | silence | distracted |
| System: | present | | System: | present | |
| User: | silence | attentive | User: | silence | distracted |
| System: | present | | System: | offer questions | |
| User: | silence | distracted | User: | ask on topic | attentive |
| System: | present | | System: | answer question | |
| User: | silence | distracted | User: | feedback | attentive |
| System: | offer questions | | System: | offer questions | |
| User: | ask on topic | attentive | User: | negate | attentive |
| System: | answer question | | System: | offer next exhibit | |
| User: | request change | attentive | User: | negate | attentive |
| System: | change exhibit | | System: | offer more | |
| User: | silence | attentive | User: | negate | attentive |
| System: | present | | System: | goodbye | |
| User: | silence | distracted | User: | goodbye | attentive |

plan to extend the dialog manager to the partially observable world, expecting better error-recovery performance and scalability.

**Acknowledgements.** This work was funded by ESF grant number 100130198.

## References

1. Faber, F., Bennewitz, M., Eppner, C., Gorog, A., Gonsior, C., Joho, D., Schreiber, M., Behnke, S.: The humanoid museum tour guide robotinho. In: Proc. 18th IEEE Int. Symp. Robot and Human Interactive Communication RO-MAN 2009 (2009)
2. Kopp, S., Gesellensetter, L., Krämer, N., Wachsmuth, I.: A conversational agent as museum guide. In: Intelligent Virtual Agents. Springer Berlin / Heidelberg (2005)
3. Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. The Journal of Machine Learning Research 4, 1107–1149 (2003)
4. Latzke, T., Behnke, S., Bennewitz, M.: Imitative reinforcement learning for soccer playing robots. In: RoboCup 2006: Robot Soccer World Cup X. Springer Berlin Heidelberg (2007)
5. Levin, E., Pieraccini, R., Eckert, W.: Using markov decision process for learning dialogue strategies. In: Acoustics, Speech and Signal Processing (1998)
6. Misu, T., Georgila, K., Leuski, A., Traum, D.: Reinforcement learning of question-answering dialogue policies for virtual museum guides. In: Proc. 13th Annual Meeting of the SIGDIAL (2012)
7. Peters, J., Schaal, S.: Natural actor-critic. Neurocomputing 71, 1180–1190 (2008)
8. Schatzmann, J., Weilhammer, K., Stuttle, M., Young, S.: A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. Knowl. Eng. Rev. 21, 97–126 (2006)
9. Thomson, B.: Statistical methods for spoken dialogue management. Ph.D. thesis, University of Cambridge, Department of Engineering (2009)

# A Framework for Optimization of Statistical Classification Measures Based on Generalized Learning Vector Quantization

Marika Kaden and Thomas Villmann

Computational Intelligence Group,
Department for Mathematics/Natural & Computer Sciences,
University of Applied Sciences Mittweida, Mittweida, Germany
email: {kaestner,villmann}@hs-mittweida.de

**Abstract.** We propose a framework for classification learning based on generalized learning vector quantization using statistical quality measures as cost function. Statistical measures like the $F$-measure or the Matthews correlation coefficient reflect better the performance for two-class classification problems than the simple accuracy, in particular if the data classes are imbalanced. For this purpose, we introduce soft approximations of those quantities contained in the confusion matrix, which are the basis for the calculation of the quality measures.

## 1 Introduction

Classification of data is one of the most frequent task in machine learning and statistical data analysis. Many methods and approaches were developed ranging from linear discriminant analysis (LDA) to classification trees. Among them, prototype based classifiers like Support Vector Machines (SVMs) or the family of Learning Vector Quantizers (LVQ) frequently yield excellent results outperforming classical statistical approaches. These approaches typically try to minimize the classification error or at least approximations thereof or, equivalently the classification accuracy.

Yet, the performance evaluation of a classifier only based on the accuracy is not the full truth. In case of imbalanced data the classification accuracy might be very high but ignoring completely underrepresented classes. This problem frequently occurs in medicine, when only a few patient data are available compared to the number of data of volunteers [4,5,22].

Therefore, evaluation of the whole confusion matrix provides much more detailed information, which can be summarized in performance values like precision, sensitivity etc. for two-class problems. These quantities are also important if the different types of misclassifications (false negatives / false positives) cause different costs [7]. We denote this scenario as an *asymmetric* classification task

(ACT). Several measures are known in statistical data analysis based on the evaluation of the confusion matrix emphasizing different aspect. Well-known are the $F$-measure, the $\chi^2$-statistics or the Jaccard-Index. Thus, a direct optimization of these quantities by a classifier model would be desirable.

Instead of direct optimization of the $F$-measure as proposed in [2], we present in this paper an approach to replace the accuracy based cost function in generalized learning vector quantization (GLVQ) by statistical measures based on the confusion matrix while keeping the idea of prototype based classification models and stochastic gradient descent learning. For this purpose, an adequate description of the confusion matrix entries in terms of the GLVQ classifier function is considered. We provide the theoretical basis to incorporate these quantities in the framework of GLVQ.

## 2 Prototype Based Classification by GLVQ

Prototype based classification models provide a powerful strategy for nonlinear classification tasks. Prominent examples are Support Vector Machines (SVMs, [20]) or Learning Vector Quantizers (LVQs, [9]). SVMs translate the classification task into a convex optimization problem whereas LVQs heuristically adjust the prototypes to optimize a Bayes decision [8]. A cost function based variant of LVQ was proposed by Sato&Yamada (*Generalized* LVQ - GLVQ, [19]. In the following we will concentrate on GLVQ.

We suppose data vectors $\mathbf{v} \in V \subseteq \mathbb{R}^n$, and the prototypes of the GLVQ model are the set $W = \{\mathbf{w}_k \in \mathbb{R}^n, k = 1 \ldots M\}$. Each data vector $\mathbf{v}$ of the training data belongs to a class $x_\mathbf{v} \in \mathcal{C} = \{1, \ldots, C\}$. The prototypes are labeled by $y_{\mathbf{w}_k} \in \mathcal{C}$. The cost function minimized by GLVQ is

$$E_{GLVQ}(W) = \frac{1}{2} \sum_{\mathbf{v} \in V} f(\mu(\mathbf{v})) \tag{1}$$

where

$$\mu(\mathbf{v}) = \frac{d^+(\mathbf{v}) - d^-(\mathbf{v})}{d^+(\mathbf{v}) + d^-(\mathbf{v})} \tag{2}$$

is the classifier function. We remark that $\mu(\mathbf{v}) \in [-1, 1]$. Further $d^+(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^+)$ denotes the dissimilarity between the data vector $\mathbf{v}$ and the closest prototype $\mathbf{w}^+$ with the same class label $y_{\mathbf{w}^+} = x_\mathbf{v}$, and $d^-(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^-)$ is the dissimilarity degree for the best matching prototype $\mathbf{w}^-$ with a class label $y_{\mathbf{w}^-}$ different from $x_\mathbf{v}$.

The dissimilarity measure $d(\mathbf{v}, \mathbf{w}_k)$ is not necessarily required to be a metric [15] but is assumed to be differentiable with respect to $\mathbf{w}_k$ for stochastic gradient learning. The transfer or squashing function $f$ is a monotonically increasing function usually chosen as sigmoid or the identity function.

Learning in GLVQ of $\mathbf{w}^+$ and $\mathbf{w}^-$ is usually performed by the *stochastic* gradient descent with respect to the cost function $E_{GLVQ}$ for a given data vector $\mathbf{v}$. Recent approaches include relational and median learning [3,13,24].

The recall for a given data point $\mathbf{v}$ is realized via a winner take all rule: Let

$$s(\mathbf{v}) = argmin_{k=1}^{M} \left( d(\mathbf{v}, \mathbf{w}_k) \right) \tag{3}$$

be the winner or best matching unit. The respective prototype label $y_{s(\mathbf{v})}$ is the predicted class of the classifier. Hence, $\mu(\mathbf{v})$ becomes negative if $\mathbf{v}$ is correctly classified, i.e. if $x_{\mathbf{v}} = y_{s(\mathbf{v})}$ is valid.

## 3 Classification Accuracy and Statistical Measures in GLVQ

### 3.1 Classification Accuracy in the GLVQ Model

Obviously, the classifier function $\mu(\mathbf{v})$ from (2) becomes negative if the data point $\mathbf{v}$ is correctly classified. Further, the transfer function $f$ is frequently chosen as the sigmoid function

$$f_\theta(x) = \frac{1}{1 + \exp\left(-\frac{x}{\theta}\right)} \tag{4}$$

with the parameter $\theta$ determining the slope [23]. In the limit $\theta \to 0$ the sigmoid $f_\theta$ becomes the Heaviside function $H(x)$, such that the cost function $E_{GLVQ}$ approximately counts the misclassifications in the GLVQ for this case. Hence, $E_{GLVQ}$ is implicitly based on the classification accuracy evaluation.

### 3.2 Statistical Measures for Classification Evaluation

Yet, accuracy is not the best choice to evaluate a classifier, in particular, if the data are imbalanced [18]. For example, assigning every object to the larger set achieves a high proportion of correct predictions, but is not generally a useful classification. In statistical analysis contingency table evaluations are well-known to deal with this problem more properly. In case of two-class problems with classes $C_+$ and $C_-$ the table contains the confusion matrix Tab. 3.2.

Several measures were developed to judge the classification quality based on the confusion matrix emphasizing different aspects. For example, the quantities *precision*

$$\pi = \frac{TP}{TP + FP} = \frac{TP}{N_+} \tag{5}$$

and *recall*

$$\rho = \frac{TP}{TP + FN} = \frac{TP}{N_+} \tag{6}$$

| labels | true | | | |
|---|---|---|---|---|
| | | $C_+$ | $C_-$ | |
| predicted | $C_+$ | $TP$ | $FP$ | $\widehat{N}_+$ |
| | $C_-$ | $FN$ | $TN$ | $\widehat{N}_-$ |
| | | $N_+$ | $N_-$ | $N$ |

**Table 1.** Confusion matrix: $TP$ - true positives, $FP$ - false positives, $TN$ - true negatives, $FN$ - false negatives, $N_\pm$- number of positive/negative data, $\widehat{N}_+$ - number of predicted positive/negative data.

are used in the widely applied $F_\beta$-measure

$$\begin{aligned} F_\beta &= \frac{\left(1+\beta^2\right)\cdot\pi\cdot\rho}{\beta^2\cdot\pi+\rho} \\ &= \frac{\left(1+\beta^2\right)\cdot TP}{(1+\beta^2)\cdot TP+\beta^2\cdot FN+FP} \end{aligned} \tag{7}$$

developed by C.J. VAN RIJSBERGEN [16]. For the common choice $\beta = 1$ it is the fraction of the harmonic and the arithmetic mean of precision and recall, i.e. $\beta$ controls the influence of both values. Another measure considering all four quantities of the confusion matrix is the *Matthews correlation coefficient*

$$MMC = \frac{TP\cdot TN - FP\cdot FN}{\sqrt{(TP+FP)\cdot(TP+FN)\cdot(TN+FP)\cdot(TN+FN)}}, \tag{8}$$

which is equivalent to the $\chi^2$-statistics for a $2\times2$ contingency table. In particular,

$$|MCC| = \sqrt{\frac{\chi^2}{N}} \tag{9}$$

is valid [11,18]. Further measures are the *Jaccard index*

$$J = \frac{TP}{FP+TP+FN} \tag{10}$$

related to the Tanimoto distances [6,17] or the *normalized* mutual information

$$IC = \frac{I}{H_{data}} \tag{11}$$

based on the mutual information [10]

$$
\begin{aligned}
I = & -H_S\left(\frac{TP}{N}, \frac{TN}{N}, \frac{FP}{N}, \frac{FN}{N}\right) \\
& -\frac{TP}{N}\log\left[\frac{TP+FP}{N}\cdot\frac{TP+FN}{N}\right] \\
& -\frac{FN}{N}\log\left[\frac{TP+FN}{N}\cdot\frac{TN+FN}{N}\right] \\
& -\frac{FP}{N}\log\left[\frac{TP+FP}{N}\cdot\frac{TN+FP}{N}\right] \\
& -\frac{TN}{N}\log\left[\frac{TN+FN}{N}\cdot\frac{TN+FP}{N}\right]
\end{aligned}
$$

with $H_S\left(\frac{TP}{N}, \frac{TN}{N}, \frac{FP}{N}, \frac{FN}{N}\right)$ is the Shannon entropy of the confusion matrix rates (probabilities) and

$$
H_S^{data} = -\frac{TP+FN}{N}\log\left[\frac{TP+FN}{N}\right] - \frac{TN+FP}{N}\log\left[\frac{TN+FP}{N}\right] \tag{12}
$$

is the Shannon entropy of the class distribution for the originally labeled data [1,21].

While there is no perfect way to evaluate the confusion matrix of true and false positives and negatives by a single number, the Matthews correlation coefficient is generally regarded as being one of the best such measures [1,21].

### 3.3 GLVQ based on Statistical Measures

As we pointed out in the previous section, there are alternatives to the simple accuracy when evaluating the classification performance, in particular for imbalanced data. In the following we propose a framework to integrate them into the GLVQ. The idea behind is to keep the basic ingredients of GLVQ, which are

- prototype based classification
- gradient descent learning
- dissimilarity based classifier function $\mu(\mathbf{v})$

At this point we restrict ourselves to the two-class scenario $\{C_+, C_-\}$ of a positive class $C_+$ with class label '$\oplus$' and a negative class $C_-$ with class label '$\ominus$'. Following the observation that the sigmoid transfer function $f_\theta$ (4) approximates the Heaviside function $H$, we consider a modified classifier function

$$
\hat{\mu}(\mathbf{v}) = f_\theta(-\mu(\mathbf{v})) \tag{13}
$$

with $\hat{\mu}(\mathbf{v}) \approx 1$ iff the data point $\mathbf{v}$ is correctly classified and $\hat{\mu}(\mathbf{v}) \approx 0$ otherwise. Now we can express all quantities of the confusion matrix in terms of the ne classifier function $\hat{\mu}(\mathbf{v})$, which yields

$$TP = \sum_{j=1}^{N} \delta_{\oplus, x_{\mathbf{v}_j}} \cdot \hat{\mu}(\mathbf{v}_j), \tag{14}$$

$$FP = \sum_{j=1}^{N} \delta_{\ominus, x_{\mathbf{v}_j}} \cdot (1 - \hat{\mu}(\mathbf{v}_j)), \tag{15}$$

$$FN = \sum_{j=1}^{N} \delta_{\oplus, x_{\mathbf{v}_j}} \cdot (1 - \hat{\mu}(\mathbf{v}_j)), \tag{16}$$

and

$$TN = \sum_{j=1}^{N} \delta_{\ominus, x_{\mathbf{v}_j}} \cdot \hat{\mu}(\mathbf{v}_j) \tag{17}$$

with $\delta_{\oplus, x_{\mathbf{v}_j}}$ is the Kronecker symbol and $\delta_{\ominus, x_{\mathbf{v}_j}} = 1 - \delta_{\oplus, x_{\mathbf{v}_j}}$. Obviously, all these quantities are differentiable with respect to $\hat{\mu}(\mathbf{v}_j)$ and, hence, also with respect to the prototypes $\mathbf{w}$, when used in stochastic gradient learning for GLVQ according to the chain rule of differentiation.

Now, we suppose without loss of generality a statistical measure $S(TP, FP, FN, TN) \in \mathcal{C}^1$ to be minimized, i.e. it is continuous and differentiable. We immediately conclude that then the function $S(TP, FP, FN, TN)$ can serve as a new cost function in the GLVQ scheme. Hence, the GLVQ can be used in a statistical framework.

Clearly, the above mentioned measures $F_\beta$, $MMC$, $J$, and $IC$ belong to this function class and, therefore, can be plugged into the GLVQ scheme.

## 4 Conclusion

In the present paper we provide a theoretical framework for GLVQ using statistical measures as cost function. The statistical measures are assumed to be continuously depending on the entries of the confusion matrix and differentiable. Then, the key idea is to use the smooth approximations of the quantities of the confusion matrix when the statistical measure is taken to replace the original accuracy based cost function in GLVQ. In this way, the basic principles of GLVQ-like prototype based classification and gradient descent learning are kept.

Thus, the new approach is an alternative to recently proposed classifier systems based on SVM and multilayer perceptron optimizing the $F_1$-objective [12,14]. Further, the general formulation allows the utilization of other statistical measures to reflect different aspects in classification learning like imbalanced class data and asymmetric classification tasks.

We presented the framework for a two-class scenario so far. Extensions to more classes could be greedy strategies like hierarchical or weighted one-versus-all classification schemes as suggested in [1]. This is topic for future research as well as the integration of such statistical measurements into fuzzy classification schemes.

## References


[1] P. Baldi, S. Brunak, Y. Chauvin, and C. Andersen H. Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424, 2000.

[2] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An exact algorithm for F-measure maximization. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, 2011.

[3] B. Hammer, D. Hofmann, F.-M. Schleif, and X. Zhu. Learning vector quantization for (dis-)similarities. *Neurocomputing*, page in press, 2013.

[4] W. Hermann, H. Barthel, S. Hesse, F. Grahmann, H.-J. Kühn, A. Wagner, and Th. Villmann. Comparison of clinical types of Wilson's disease and glucose metabolism in extrapyramidal motor brain regions. *Journal of Neurology*, 249(7):896–901, 2002.

[5] W. Hermann, P. Günther, A. Wagner, and T. Villmann. Klassifikation des Morbus Wilson auf der Basis neurophysiologischer Parameter. *Der Nervenarzt*, 76:733–739, 2005.

[6] P. Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11:37–50, 1912.

[7] M. Kästner, W. Hermann, and T. Villmann. Integration of structural expert knowledge about classes for classification using the fuzzy supervised neural gas. In M. Verleysen, editor, *Proc. of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'2012)*, pages 209–214, Louvain-La-Neuve, Belgium, 2012. i6doc.com.

[8] Teuvo Kohonen. Learning Vector Quantization. *Neural Networks*, 1(Supplement 1):303, 1988.

[9] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 1995. (Second Extended Edition 1997).

[10] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[11] B.W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta*, 405:442–451, 1975.

[12] D.R. Musicant, V. Kumar, and A. Ozgur. Optimizing F-measure with support vector machines. In *PROCEEDINGS OF THE SIXTEENTH INTERNATIONAL FLORIDA ARTIFICIAL INTELLIGENCE RESEARCH SOCIETY CONFERENCE*, pages 356–360. Haller AAAI Press, 2003.

[13] D. Nebel and T. Villmann. A median variant of generalized learning vector quantization. In *Proceedings of International Conference on Neural Information Processing (ICONIP)*, page submitted. Springer, 2013.

[14] J. Pastor-Pellicer, F. Zamora-Martínez, S. España-Boquera, and M.J. Castro-Bleda. F-measure as the error function to train neural networks. In I. Rojas, G. Joya, and J. Gabestany, editors, *Advances in Computational Intelligence - 12th International Work-Conference on Artificial Neural Networks, IWANN, Puerto de la Cruz, Tenerife, Spain*, volume 1 of *LNCS*, pages 376–384, 2013.

[15] E. Pekalska and R.P.W. Duin. *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications*. World Scientific, 2006.

[16] C.J. Rijsbergen. *Information Retrieval*. Butterworths, London, 2nd edition edition, 1979.

[17] D. J. Rogers and T.T. Tanimoto. A computer program for classifying plants. *Science*, 132(3434):1115–1118, 1960.

[18] L. Sachs. *Angewandte Statistik*. Springer Verlag, 7-th edition, 1992.

[19] A. Sato and K. Yamada. Generalized learning vector quantization. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8. Proceedings of the 1995 Conference*, pages 423–9. MIT Press, Cambridge, MA, USA, 1996.

[20] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

[21] C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–432, 1948.

[22] T. Villmann, G. Blaser, A. Körner, and C. Albani. Relevanzlernen und statistische Diskriminanzverfahren zur ICD-10 Klassifizierung von SCL90-Patienten-Profilen bei Therapiebeginn. In G. Plöttner, editor, *Aktuelle Entwicklungen in der Psychotherapieforschung*, pages 99–118. Leipziger Universitätsverlag, Leipzig, Germany, 2004.

[23] A.W. Witoelar, A. Gosh, J.J. de Vries, B. Hammer, and M. Biehl. Window-based example selection in learning vector quantization. *Neural Computation*, 22(11):2924–2961, 2010.

[24] X. Zhu, F.-M. Schleif, and B. Hammer. Semi-supervised vector quantization for proximity data. In M. Verleysen, editor, *Proc. of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'2013)*, pages 89–94, Louvain-La-Neuve, Belgium, 2013. i6doc.com.

# Classifier inspection based on different discriminative dimensionality reductions

Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer

University of Bielefeld - CITEC centre of excellence, Germany
{aschulz|agisbrec|bhammer}@techfak.uni-bielefeld.de

**Abstract.** Nonlinear dimensionality reduction (DR) techniques offer the possibility to visually inspect a high-dimensional data set in two dimensions, and such methods have recently been extended to also visualize class boundaries as induced by a trained classifier on the data. In this contribution, we investigate the effect of two different ways to shape the involved dimensionality reduction technique in a discriminative way: discrimination based on the data labels of the given data set, and discrimination based on the labels as provided by the trained classifier. We demonstrate that these two different techniques lead to semantically different visualizations which allow us to further inspect the classification behavior. Both approaches can uniformly be based on the Fisher information matrix, which is estimated in two different ways.

**Keywords:** Visualization of Classifiers, Supervised Dimensionality Reduction, Fisher Information

## 1 Introduction

Big data as well as complex settings cause the need of humans to interactively process and interpret large, heterogeneous, high-dimensional data sets, specifying the learning goals and appropriate data analysis tools based on obtained findings [22, 8, 17]. Here, interpretability of the models and nonlinear data visualization play a major role [20, 10, 14]. A classifier is not only judged by its classification accuracy, rather, it moves into the focus which decision the classifier is taking and why. In this context, it becomes interesting not only to visualize the given data sets, but also the classifier inferred thereof. This possibility allows us to extract information beyond the mere classification accuracy addressing questions such as: are there data which are observable as outliers and hence are potentially mis-labeled, are there noisy data regions where the classification is inherently difficult, are there regions where the flexibility of the classifier is not yet sufficient, what is the modality of single classes, etc.

At present, visualization in the context of classifiers is often restricted to interfaces to set certain parameters: Brier [7] and ROC curves plot a point for each setting of a certain parameter and, such, allow to choose the best setting for a certain purpose. There exists relatively little work to visualize the underlying classifier itself. Such include tour methods [3] allowing to interactively

choose linear projections; nomograms [9] analyzing certainty measures provided by the classifier, or linear projection techniques on top of this measures [13]. A few nonlinear techniques exist such as SVMV [21] where a SOM is utilized as a projection method. Recently, a general framework how to visualize a general classifier based on nonlinear dimensionality reduction techniques has been proposed [15], which enables to project any given classifier and underlying data points to two dimensions. One key aspect of this framework is the integration of discriminative dimensionality reduction rather than classical DR methods, such that conflicts arising from the projection of high-dimensional data are solved by focussing on the discriminative dimensions of the data only.

In this contribution, we investigate two different ways to estimate discriminative information of the data; in particular, we distinguish two different sources of this information: the underlying data labels of a given training set, and the labeling provided by the classifier itself. We demonstrate that these two techniques lead to two computationally different ways to estimate discriminative information which have a strong influence on the information shown in the visualization: we mainly see aspects of the data manifold for the first setting, while we can have a view on the way the classifier labels the data in the second approach. We demonstrate this difference in a first illustrative scenario.

Now we first shortly review the general framework how to visualize a given classifier, and include discriminative dimensionality reduction based on the Fisher information, afterwards. As a novel contribution, we explain two different ways to estimate the Fisher information, and we demonstrate the effect in an experiment.

## 2 The general framework

This section follows the description as provided in [15]. We assume the following scenario: a finite data set of points $\mathbf{x}_i \in X = \mathbb{R}^n$ and labeling $l_i \in L$ is given. Furthermore, a classifier $f : X \to L$ has been trained on these data. which generates a label $f(\mathbf{x})$. We assume that these labels are accompanied by a real value $r(\mathbf{x}) \in \mathbb{R}$ which is a monotonic function depending on the minimum distance from the class boundary such as provided by most classification techniques.
The framework consists of the following three steps:

(I) Project the data $\mathbf{x}_i$ using a DR technique leading to points $\pi(\mathbf{x}_i) \in Y = \mathbb{R}^2$.
(II) Sample the projection space $Y$ leading to points $\mathbf{z}'_i$. Determine points $\mathbf{z}_i$ in the data space $X$ which correspond to these projections $\pi(\mathbf{z}_i) \approx \mathbf{z}'_i$.
(III) Visualize the training points $\mathbf{x}_i$ together with the contours induced by the sampled function $(\mathbf{z}'_i, |r(\mathbf{z}_i)|)$.

Note that this procedure avoids problems which would be caused by a high dimensionality of the data by two ingredients: DR of the given data points only focuses on aspects defined by the data distribution, this way avoiding the problem to correctly visualize the full input space in two dimensions which is obviously not possible. Further, sampling of the feasible region to show the class boundaries takes place in the low-dimensional projection space only, avoiding computational intractability of sampling in the data space itself.

Two questions remain open in this abstract framework: How can we determine inverse points $\mathbf{z}_i$ for given projections $\mathbf{z}'_i$ which correspond to inverse images in the data manifold? What properties should the DR technique fulfill such that only the important aspects of the setting are visualized? In this paper, we will focus on the latter question and briefly discuss the first one in the next section based on the description in [15].

In order to quantitatively evaluate the obtained visualization of the classifier, we calculate the accordance between the original classification and the visualized one: We compute the predicted label of the classifier $f(\mathbf{x}_i)$ for those points $\mathbf{x}_i$ which have not been used to optimize the inverse mapping $\pi^{-1}$ (see section 3). Then we calculate the label which would be assigned by the visualized classifier to the low-dimensional counterparts of these points $\mathbf{x}_i$. Calculating the agreement of these labels gives a measure of the quality of the visualized classifier.

## 3 Inverse nonlinear dimensionality reduction

Given a nonlinear projection of points $\mathbf{x}_i \in X$ to $\pi(\mathbf{x}_i) = \mathbf{y}_i \in \mathbb{R}^2$ and additional data points $\mathbf{z}'_i \in \mathbb{R}^2$, what are points $\mathbf{z}_i$ such that its projections approximate $\mathbf{z}'_i \approx \pi(\mathbf{z}_i)$ and, in addition, $\mathbf{z}_i$ are contained in the data manifold?

Following [15], we use an interpolation technique with the general form

$$\pi^{-1} : Y \to X, \mathbf{y} \mapsto \frac{\sum_i \boldsymbol{\alpha}_i k(\mathbf{y}_i, \mathbf{y})}{\sum_i k(\mathbf{y}_i, \mathbf{y})} = \mathbf{A}\mathbf{k}, \tag{1}$$

where the parameters $\boldsymbol{\alpha}_i \in X$ can be interpreted as prototypical preimages of $\mathbf{y}_i$ for small bandwidths $\sigma$ of the Gaussian kernel $k(\mathbf{y}_i, \mathbf{y}) = \exp(-0.5\|\mathbf{y}_i - \mathbf{y}\|^2/\sigma^2)$. $\sigma$ is chosen as $c \cdot E_{\mathbf{x}_i}\{\min_{j \neq i} \|\mathbf{x}_i - \mathbf{x}_j\|\}$ where $c$ is a predefined scaling factor. The sum is either over a subset of $m$ given data projections $\mathbf{y}_i = \pi(\mathbf{x}_i)$, or over $m$ codebooks representing the $\mathbf{y}_i$. The matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ contains the $\boldsymbol{\alpha}_i$ in its columns and $\mathbf{k} \in \mathbb{R}^{m \times 1}$ is a vector of normalized kernel values. The normalization yields that the projected data points lie in the convex hull of the $\boldsymbol{\alpha}_i$. The mapping $\pi^{-1}$ is trained on the points $(\mathbf{x}_i, \pi(\mathbf{x}_i))$ corresponding to the data manifold $X$ only. Due to this training set, an inversion of the projection $\pi$ is emphasized which maps points in $Y$ to inverse points which lie in the original data manifold, and ambiguities of the ill-posed problem to map the low-dimensional projection space to the full high-dimensional data space are resolved.

The parameters $\boldsymbol{\alpha}_i$ can be obtained by minimizing the following supervised error of the projection $\pi^{-1}$ on the training set

$$E = \sum_i d^2_{\mathbf{J}}\left(\mathbf{x}_i, \pi^{-1}(\mathbf{y}_i)\right) = \sum_i \left(\mathbf{x}_i - \pi^{-1}(\mathbf{y}_i)\right)^\top \mathbf{J}(\mathbf{x}_i) \left(\mathbf{x}_i - \pi^{-1}(\mathbf{y}_i)\right). \tag{2}$$

The matrix $\mathbf{J}$ refers to the Fisher information matrix as defined in section 4. In contrast to a standard Euclidean error function, this function has the advantage that those dimensions in $X$ which are locally relevant for the classification are emphasized. Minimization of these costs takes place by gradient techniques.

In contrasts to the error function used in [15], this function consists only of one term such that a balance of an unsupervised and a supervised term does not need to be found (this has already been done during the computation of $\mathbf{J}$).

## 4 Discriminative nonlinear visualization

The general framework yields reasonable results provided the data manifold is intrinsically low-dimensional and the DR method is capable of taking this information into account. For data manifolds with intrinsic dimensionality larger than two, a DR to two dimensions necessarily disrupts potentially relevant information. Due to this fact, it has been proposed in [15] to use a discriminative dimensionality reduction technique in step (I) which allows the user to explicitly specify which aspects of the data are regarded as relevant and which aspects of the data can be neglected in the considered setting.

A variety of different discriminative DR techniques has been proposed, such as Fisher's linear discriminant analysis (LDA), partial least squares regression (PLS), informed projections [4], global linear transformations of the metric [6, 2], or kernelization of such approaches [11, 1]. A rather general idea to include supervision is to locally modify the metric [12, 5] by defining a Riemannian manifold which takes into account auxiliary information of the data and which measures the effect of directions of the data on this auxiliary information. This modified metric can then be plugged into many modern DR techniques such as t-SNE.

In our case, we assume that the auxiliary information is provided by discrete class information $c$ assigned to $\mathbf{x}$, where $c$ captures aspects relevant for the class labeling, which should be visualized. This information can locally be incorporated into the distance computation by setting the quadratic form of the tangential space at $\mathbf{x}$ as $d^2_{\mathbf{J}}(\mathbf{x}, \mathbf{x}+d\mathbf{x}) = (d\mathbf{x})^\top \mathbf{J}(\mathbf{x})(d\mathbf{x})$, where $\mathbf{J}(\mathbf{x})$ is the local Fisher information matrix

$$\mathbf{J}(\mathbf{x}) = E_{p(c|\mathbf{x})}\left\{\left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x})\right)\left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x})\right)^\top\right\}. \tag{3}$$

Thereby, $p(c|\mathbf{x})$ denotes the probability of the class information $c$ conditioned on $\mathbf{x}$. These local distances can be extended to the entire manifold by taking minimum path integrals.

Since exact minimization and integration is usually computationally intractable, approximations are used; a description of approximations with their corresponding advantages and disadvantages can be found in [12]. In the following experiments we will use linear piecewise distance approximations.

The Fisher information matrix can be directly included into all distance based techniques. In this paper we integrate it into the popular non-parametric DR method t-Distributed Stochastic Neighbor Embedding (t-SNE). This method projects data points such that the probabilities of data pairs are preserved in the low-dimensional space [19]. A Gaussian distribution is assumed in the high-dimensional space and a Student-t distribution in the low-dimensional space, addressing the crowding problem. We will refer to Fisher t-SNE when we replace the Euclidean metric with the Fisher metric in t-SNE.

Due to the supervised selection of information in the high-dimensional space, the variability of projections obtained by Fisher t-SNE is less as compared to those by t-SNE.

## 5 Estimating the Fisher information matrix

The central part of this modified distance computation consists in the estimation of the probability $p(c|\mathbf{x})$ of information $c$ given a data point $\mathbf{x}$. In our setting, there are two essentially different possibilities how to choose this information:

(a) We can use the given class labels $c := l$ or $c_i := l_i$ for data point $\mathbf{x}_i$, respectively, as provided in the training set. This choice emphasizes the given 'ground truth' of the data, and, in consequence, the visualization of the classifier visualizes in which regions the obtained classification is simple or complex as compared to this ground truth.

(b) We can use the labeling as provided by the trained classifier $c := f(\mathbf{x})$. This choice emphasizes aspects of the data which are regarded by the classifier as interesting. Hence, those aspects of the data are visualized which influence the trained classification; as an example one can detect regions of the data where points are regarded as virtually identical by the classifier.

Apart from the different semantic meaning, this choice has consequences on the possibilities how to compute the probability $p(c|\mathbf{x})$. The Fisher matrix is based on the local change of the probability distribution $p(c|\mathbf{x})$, the latter of which is usually unknown and needs to be approximated. A common way to do this is to use the Parzen window non-parametric estimator as proposed in [12]. This yields a correct estimation of the probability density but is computationally expensive, i.e. $\mathcal{O}(N^2)$ for $N$ data points, and, for finite data sets, the result depends on the chosen bandwidth[1] of the estimator. The resulting estimator is differentiable and the derivatives are reported in [12], for example. As an alternative, the authors in [12] suggest to use faster methods to approximate $p(c|\mathbf{x})$ which are less accurate, however. Essentially, these methods substitute the non-parametric estimator by a parametric one such as a Gaussian mixture model, the parameters of which are optimized on the data. Hence, the techniques require an additional computational step to extract a model which approximates $p(c|\mathbf{x})$.

For the choice of $c := f(\mathbf{x})$, a more direct method can be used. The classifier itself provides a model for $f(\mathbf{x})$, and, often, this output or its continuous counterpart $r(\mathbf{x})$ also naturally provide a probabilistic model for $p(f(\mathbf{x})|\mathbf{x})$. Besides the computational advantage of no longer having the need to train an additional model, this alternative is also parameterless. Probabilities can be extracted from the value $r(\mathbf{x})$ for popular models such as SVM, see e.g. [23], alternatives such as robust soft learning vector quantization [16] directly train a generative model for classification. In all cases, derivatives can either be determined analytically or numerically e.g. simply using a linear approximation of the derivatives.

## 6 Experiments

In this section we illustrate the difference between estimating the probability distribution $p(c|\mathbf{x})$ from the Parzen window estimator for the given labeling

[1] We use the estimator $\hat{h}_{rot}$ provided in the literature to specify this parameter, see e.g. [18].

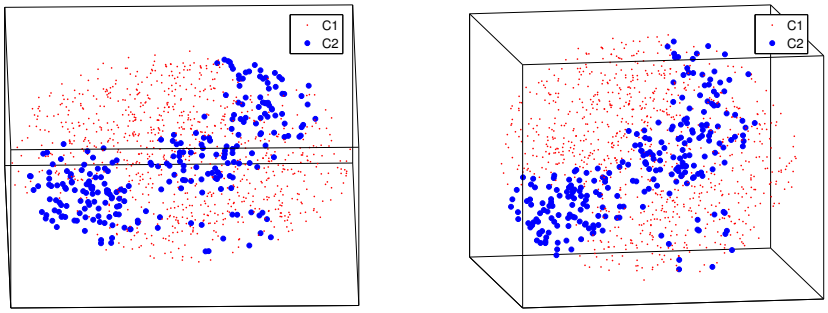



**Fig. 1.** The three-dimensional data set shown from two different perspectives.

$c := l$ and estimating $p(c|\mathbf{x})$ from the trained classification model $c := f(\mathbf{x})$ using a direct computation of the gradient based on the classifier $f$. For this purpose, we create an artificial data set which is intrinsically three-dimensional and, hence, cannot be projected to two dimensions without information loss. The data are arranged in a filled ball with a nonlinear class structure. The data set is shown in Fig. 1 from two perspectives. The blue class consists of a continuous tube with one gap and a noisy region.

An unsupervised projection of this data set with t-SNE is shown in Fig. 2. The projection distorts the continuous class structure. This constitutes an obvious result since the data distribution is uniform and such doesn't resemble the class structure. It illustrates that unsupervised visualization techniques might not always be well suited if intrinsically high-dimensional data should be projected to low dimensions. In this example, the displayed information looks almost arbitrary.

Now we train the classifier Robust Soft LVQ (RSLVQ)[16] on this data set. This classification scheme learns a prototype based probabilistic model for the data such that the likelihood of correct classification is optimized. The resulting probability functions of the classifier are differentiable and, therefore, can be easily integrated into the Fisher metric framework discussed in section 4. We use four prototypes per class, which is small considering the complexity of the data set. The trained classifier achieves a classification accuracy of 90%. Now, a typical use case for the classifier visualization method occurs: How did the classification method solve this problem? Which simplifications of the data did the classifier use and which data points are regarded as similar by the classifier?

In order to answer these questions we visualize the classifier using the original class labels $l_i$ on the one hand and using the provided classification $f(\mathbf{x})$ on the other hand. We build the visualization of the classifier on top of these two projections. The two resulting visualizations are depicted in Fig. 3. The left visualization is based on the Parzen window estimator for the class labels $p(l|\mathbf{x})$: Basically, two clusters of points from class blue are shown and these are distinct

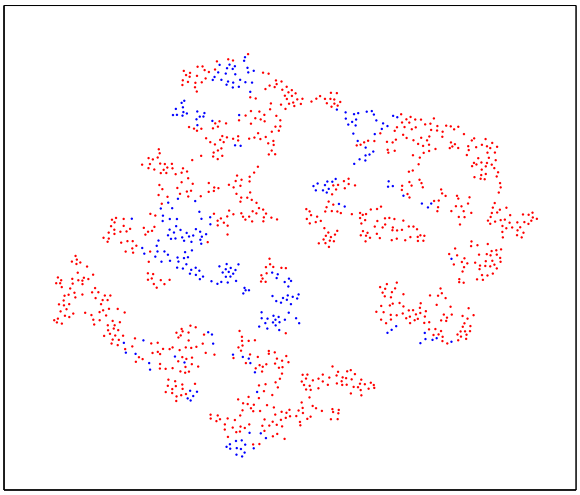


**Fig. 2.** Projected data with t-SNE.

from each other. The visualization quality of the classifier amounts to 92%. Interestingly, albeit this is not yet perfect, the visualization looks much more reasonable than direct unsupervised t-SNE on the data. The right visualization shows the same classifier, but this time based on the discriminative projection obtained by using the probabilities of the classifier itself $p(f(\mathbf{x})|\mathbf{x})$. The data from class two form again two clusters, but this time, they are close to each other. The quality is estimated to 95%. Furthermore, the shape of the class boundaries resembles more the expected shape of the classifier, the latter usually being related to convex regions. In the visualization based on the ground truth, the original spherical shape of the data is much more pronounced.

The Parzen window estimator used in the left visualization estimates the probability density accurately and finds the gap in the blue class tube. In his part of the data space, the class distribution changes rapidly and, therefore, the distances in this region grow large, which can directly be observed in the visualization. The prototype distribution does not fit very well to the visualized classifier, since in one region of the blue class there are three prototypes of that class on top of each other and in another region there are none. But since the visualized class distribution is correct as concerns a large part of the points, we can see from this visualization that the largest part of the blue class tube is classified correctly.

In the right visualization which is based on the classifier probabilities, the two parts of the tube lie close together. This implies that in the probability density of the classifier the class distribution does not change much, i.e. the data

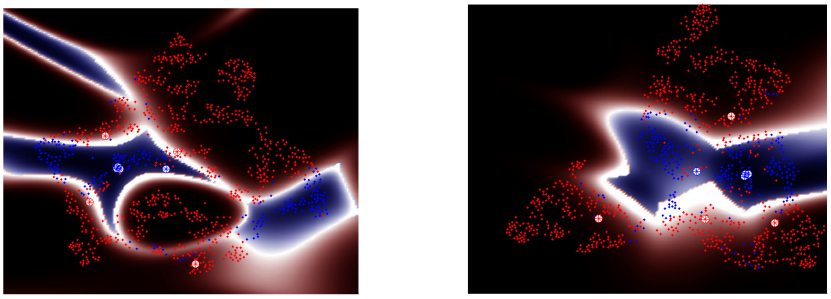


**Fig. 3.** Two visualization of the same RSLVQ classification model: The projection methods Fisher t-SNE based on the Parzen window estimator (left) and Fisher t-SNE based on the probabilities from the trained classifier (right) are applied.

lying in this gap of the tube are classified incorrectly. However, only because this information is included in the projection via the class probability $p(f(\mathbf{x})|\mathbf{x})$, this fact can be seen directly in the visualization. This time, the location of the prototypes is plausible in relation to the data: the prototypes of the blue class are surrounded by those of the red class. Such a constellation is plausible in the original data space.

From the latter visualization we can deduce more information as regards potential errors as compared to the previous one; we see directly the source of the remaining classification error: the classifier is not powerful enough and is not able to classify this gap in the data correctly. Furthermore, there are a few points from the blue class which lie in the cluster of points from the red class. From the perspective of this visualization we would deduce that these are either overlapping regions or too complex regions for our classifier (both aspects are probably correct: in the high-dimensional data we can see that there is indeed a region of overlapping classes).

For this toy example we can verify our interpretation by visualizing the positions of the prototypes in the original data space. Fig. 4 depicts the original data set in conjunction with the prototypes of the classifier. The same positioning of the prototypes as in the low-dimensional visualization emerges: the prototypes of the blue class are surrounded by those of the red class.

## 7 Discussion

In this contribution, we have proposed two different ways how to integrate auxiliary class labeling into a classifier visualization, and we discussed the different semantic meaning of the two techniques. We demonstrated in a simple toy example that a visualization based on a classifier not only eases the computational burden of density estimation for the Fisher metric, but also gives a clearer picture

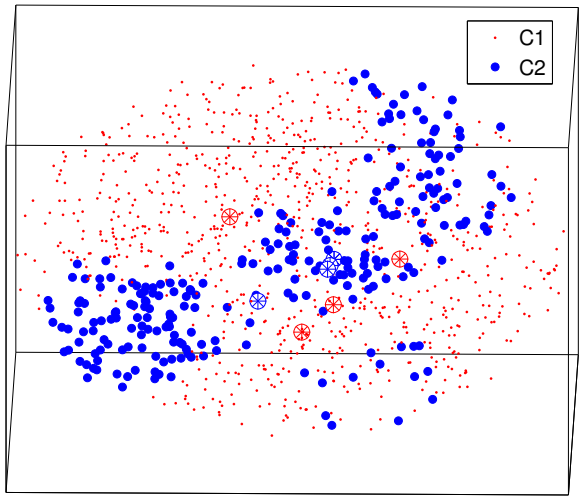



**Fig. 4.** Projected data with t-SNE.

about the classification principles as quantitatively evaluated by the accordance, and as discussed by focusing on specific aspect of the data.

This preliminary research opens the way towards a more thorough evaluation of the technique in benchmark examples and efforts to turn it to a ready-to-use visualization tool for the behavior of classifiers for high-dimensional data sets.

## ACKNOWLEDGEMENTS

Funding from DFG under grant number HA2719/7-1 and by the CITEC centre of excellence is gratefully acknowledged.

## References

1. G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12:2385–2404, 2000.
2. K. Bunte, P. Schneider, B. Hammer, F.-M. Schleif, T. Villmann, and M. Biehl. Limited rank matrix learning, discriminative dimension reduction and visualization. *Neural Networks*, 26:159–173, 2012.
3. D. Caragea, D. Cook, H. Wickham, and V. Honavar. Visual methods for examining svm classifiers. In Simoff et al. [17], pages 136–153.
4. D.Cohn. Informed projections. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 849–856. MIT Press, 2003.

5. A. Gisbrecht, D. Hofmann, and B. Hammer. Discriminative dimensionality reduction mappings, 2012.
6. J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2004.
7. J. Hernandez-Orallo, P. Flach, and C. Ferri. Brier curves: a new cost-based visualisation of classifier performance. In *International Conference on Machine Learning*, June 2011.
8. T. W. House. Big data research and development initiative, 2012.
9. A. Jakulin, M. Možina, J. Demšar, I. Bratko, and B. Zupan. Nomograms for visualizing support vector machines. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 108–117, New York, NY, USA, 2005. ACM.
10. J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, 2007.
11. B. Ma, H. Qu, and H. Wong. Kernel clustering-based discriminant analysis. *Pattern Recognition*, 40(1):324–327, 2007.
12. J. Peltonen, A. Klami, and S. Kaski. Improved learning of riemannian metrics for exploratory analysis. *Neural Networks*, 17:1087–1100, 2004.
13. F. Poulet. Visual svm. In C.-S. Chen, J. Filipe, I. Seruca, and J. Cordeiro, editors, *ICEIS (2)*, pages 309–314, 2005.
14. S. Rüping. *Learning Interpretable Models*. PhD thesis, Dortmund University, 2006.
15. A. Schulz, A. Gisbrecht, and B. Hammer. Using nonlinear dimensionality reduction to visualize classifiers. In I. Rojas, G. J. Caparrós, and J. Cabestany, editors, *IWANN (1)*, volume 7902 of *Lecture Notes in Computer Science*, pages 59–68. Springer, 2013.
16. S. Seo and K. Obermayer. Soft learning vector quantization. *Neural Computation*, 15(7):1589–1604, 2003.
17. S. J. Simoff, M. H. Böhlen, and A. Mazeika, editors. *Visual Data Mining - Theory, Techniques and Tools for Visual Analytics*, volume 4404 of *Lecture Notes in Computer Science*. Springer, 2008.
18. B. A. Turlach. Bandwidth Selection in Kernel Density Estimation: A Review. In *CORE and Institut de Statistique*, pages 23–493, 1993.
19. L. van der Maaten and G. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.
20. A. Vellido, J. Martin-Guerroro, and P. Lisboa. Making machine learning models interpretable. In *ESANN'12*, 2012.
21. X. Wang, S. Wu, X. Wang, and Q. Li. Svmv - a novel algorithm for the visualization of svm classification results. In J. Wang, Z. Yi, J. Zurada, B.-L. Lu, and H. Yin, editors, *Advances in Neural Networks - ISNN 2006*, volume 3971 of *Lecture Notes in Computer Science*, pages 968–973. Springer Berlin / Heidelberg, 2006.
22. M. Ward, G. Grinstein, and D. A. Keim. *Interactive Data Visualization: Foundations, Techniques, and Application*. A. K. Peters, Ltd, 2010.
23. T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *J. Mach. Learn. Res.*, 5:975–1005, Dec. 2004.

# Learning the Appropriate Model Population Structures for Locally Weighted Regression

Slobodan Vukanović, Alexander Schulz, Robert Haschke, and Helge Ritter

Cognitive Interaction Technology – Center of Excellence (CITEC)
Bielefeld University, Bielefeld, Germany

**Abstract.** Local learning approches subdivide the learning space into regions to be approximated locally by linear models. An arrangement of regions that conforms to the structure of the target function can lead to learning with fewer resources and gives an insight into the function being approximated. This paper introduces a covariance–based update for the size and shape of each local region that is able to exploit linear substructures in the target function. Initial tests show that the method improves the structuring capabilities of state–of–the–art statistics–based local learners, producing model populations that are shaped appropriately to the functions being approximated.

**Keywords:** Function Approximation, Locally Weighted Regression, Input Space Structuring

## 1 Introduction

Locally weighted regression (LWR) approximates the target function with a population of linear models, each localized in the input space. The output to a given query is computed as a weighted (usually Gaussian) combination of the outputs of the local models (Atkeson et al. [1997]). The principal advantage that LWR offers over classical function approximation is its independence from the choice of the class of the approximating function. This leads to learning systems that are robust to destructive interference, and whose complexity can be increased incrementally by adding new models. LWR is thus popular for learning in online scenarios, such as the ones often encountered in robotics (Sigaud et al. [2011]).

The locality of each model is defined by its position in the input space and a distance metric, which is usually adapted as the learning progresses. The adaption mechanism often aims to structure the input space by exploiting the linear subspaces that are local to each model (Ormoneit and Hastie [1999]). A reasonable structuring of the input space offers insights into the structure of the function being approximated, and it may enable simpler functions to be learned with fewer models.

It has been demonstrated (Stalph et al. [2010]) that the statistics–based Locally Weighted Projection Regression (LWPR, Vijayakumar et al. [2005]), a popular local learning system (Sigaud et al. [2011]) is not capable of structuring

the input space well. LWPR produces tight–fitting, highly non–overlapping spherical populations from which the structure is difficult to grasp.

This paper describes a work in progress[1] on improving the structuring capabilities of statistics–based LWR. We introduce a weighted covariance distance metric update for approaches that utilize ellipsoidal neighborhoods, such as LWPR and Local Linear Maps (LLM, Ritter et al. [1992]), a local learner that does not aim to structure the input space. Training samples are weighted according to their current local prediction error and the proximity to each model, and are either included in the distance metric update or ignored. The initial tests show that such an update improves the structuring over the existing distance metric updates, producing populations that visually resemble the target function. Additionally, the tests show that the improved structuring obtained by our method can lead to reductions in population sizes and thus the number of resources necessary for accurate learning.

The following section gives an overview of the distance metric adaption in LWPR and LLM. Section 3 describes our weighted covariance distance metric update method. Section 4 reports the results of an initial comparison between our approach and the default distance metric updates in LWPR. Section 5 concludes.

## 2 Distance Metric Adaption in LWPR and LLM

LWPR, a descendant of the Receptive Field Weighted Regression (RFWR, Schaal and Atkeson [1997]), performs input dimensionality reduction through partial least squares (PLS) regression. The population size is incrementally increased by allocating models in parts of the input space not covered by existing models. The pseudocode is given in Algorithm 1.

**Algorithm 1** The LWPR pseudocode. The PLS regression update (line 5) is beyond the scope of this paper (details can be found in Vijayakumar et al. [2005]).

```
1: Initialize LWPR with no local models
2: for a new training sample (x, y) do
3:    for each local model M do
4:       Calculate the activation of M by (x, y)
5:       Update the regression of M
6:       Update the distance metric D of M (Equation 2)
7:    end for
8:    if no model was activated by more than the w_gen parameter then
9:       Create a new model centered at x and initialize it using the default parameters
10:   end if
11: end for
```

[1] Parts of which have been submitted to the Thirteenth International Conference on Advances in Artificial Intelligence, AI*IA 2013.

The *receptive field* (RF) of each local model is described by an ellipsoid fixed at a point $\mathbf{c}$ in the input space and a distance metric $\mathbf{D}$, a positive semi–definite matrix:

$$(\mathbf{x}-\mathbf{c})^T\mathbf{D}(\mathbf{x}-\mathbf{c}) = 1 \tag{1}$$

A function of $\mathbf{c}$, $\mathbf{D}$, and an incoming training sample returns an *activation weight* $w \in [0,1]$ for that sample. The higher the activation $w$, the more influence the sample has on the model during training. To try to adapt a receptive field's shape and size to the local linear region, $\mathbf{D}$ is Cholesky–decomposed into $\mathbf{M}$, which is then trained by minimizing the cost function $J$ of the leave–one–out cross–validation error for each iteration step $t$ and a given learning rate $\alpha$:

$$\mathbf{M}_t = \mathbf{M}_{t-1} + \alpha\frac{\partial J}{\partial \mathbf{M}_{t-1}}, \mathbf{D}_t = \mathbf{M}_t^T\mathbf{M}_t \tag{2}$$

Unlike LWPR, the default LLM implementation uses a fixed number of models whose positions in the input space can be adapted during training, e.g. by vector quantization. The local models learn to approximate the underlying function using ordinary least squares regression. The default distance metric update adapts each receptive field along the input dimensions, increasing the coverage of the input space:

$$\mathbf{d}_t = \mathbf{d}_{t-1} + \alpha(||\mathbf{x}_t - \mathbf{c}_t|| - \mathbf{d}_{t-1}), \mathbf{D}_t = \mathrm{diag}[\mathbf{d}_t]^{-1} \tag{3}$$

As no information about the output space is given to the update, the populations of the models after training do not resemble the structure of the target function.

## 3 The Covariance–based Distance Metric Update

We consider a representation of the receptive fields identical to LWPR (Equation 1). Given a training sample $(\mathbf{x}_t, \mathbf{y}_t)$ at step $t$, we model the distance metric $\mathbf{D}_t$ of each receptive field as the inverse of the incremental covariance $\mathbf{\Sigma}_t$:

$$\mathbf{D}_t = (\mathbf{\Sigma}_t)^{-1}, \mathbf{\Sigma}_t = \frac{\mathbf{S}_t}{\Phi_t} \tag{4}$$

$\mathbf{S}_t$ is the matrix of weighted incremental sums (employing the stable covariance update from Finch [2009]):

$$\mathbf{S}_t = (1-\gamma)\cdot\mathbf{S}_{t-1} + \gamma\cdot\phi(\mathbf{p})\cdot(\mathbf{x}_t-\mathbf{c})(\mathbf{x}_t-\mathbf{c})^T \tag{5}$$

where $\gamma \in [0,1]$ is the forgetting factor, $\phi(\mathbf{p})$ is the weighting function with parameters $\mathbf{p}$, and $\mathbf{c}$ is the center of the receptive field. $\Phi_t$ is the sum of weights:

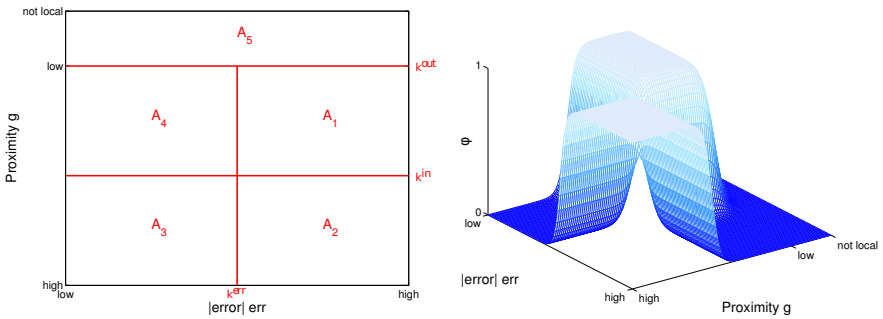
$$\Phi_t = (1-\gamma)\cdot\Phi_{t-1} + \gamma\cdot\phi(\mathbf{p}) \tag{6}$$

The weighting function $\phi$ should be such that the local training samples that produce a small error are included in the covariance update, and the local samples that produce a large error are excluded. The derivation of $\phi$ with these characteristics is given below, followed by a brief discussion on the values of the parameter vector $\mathbf{p}$.

### 3.1 The Weighting Function and its Parameters

The covariance update will prompt a receptive field to shrink to include samples that have large weights assigned by $\phi$ and a high activation $w$. Similarly, a receptive field will grow to include samples that have large weights and a low $w$. The samples with very small weights will be excluded from the covariance update regardless of their location, prompting no change in the distance metric. These dynamics must be taken into account for weight assignment: $\phi$ needs to be a function of both the current local prediction error and the activation.

Given the magnitude of the error err (low or high) and the proximity $g = 1-w$ of a training sample to the center of the field (high, low, or not local), we identify five cases $A_1, \ldots, A_5$ (shown in Figure 1(a)) to which that sample can belong. The cases $A_1, \ldots, A_4$ are modeled by functions $A_1(\mathbf{p}_1), \ldots, A_4(\mathbf{p}_4)$ that have the value 1 in the corresponding regions. The weighting function $\phi$ is written as a combination of $A_1(\mathbf{p}_1), \ldots, A_4(\mathbf{p}_4)$ multiplied by $A_5(\mathbf{p}_5)$, which has the value 0 in the region $A_5$. Figure 1(b) shows a weighting function where $A_1 = A_3 = A_5 = 0$ and $A_2 = A_4 = 1$.




(a) (b)

**Fig. 1. (a) Regions in which a training sample may fall: high error and low proximity ($A_1$), high error and high proximity ($A_2$), low error and high proximity ($A_3$), low error and low proximity ($A_4$), or not local ($A_5$). The parameters $k^{\mathbf{err}}, k^{\mathbf{in}}$, and $k^{\mathbf{out}}$ control the extents of the regions. (b) A weighting function $\phi$ with $A_1 = A_3 = A_5 = 0$ and $A_2 = A_4 = 1$.**

Let $\sigma^+$ and $\sigma^-$ be one–dimensional logistic functions:

$$\begin{aligned} \sigma^+(x, s, k) &= \frac{1}{1+e^{-s(|x|-k)}} \\ \sigma^-(x, s, k) &= \frac{-1}{1+e^{-s(|x|-k)}} + 1 \end{aligned} \tag{7}$$

where $s$ is the slope and $k$ is the position of each sigmoid. The functions $A_i(\mathbf{p}_i)$ with parameters $\mathbf{p}_i$ are products of the sigmoids:

$$\begin{aligned}
A_1(\text{err}, s_1^{\text{err}}, k_1^{\text{err}}, g, s_1^g, k_1^g) &= \sigma^+(\text{err}, s_1^{\text{err}}, k_1^{\text{err}})\sigma^+(g, s_1^g, k_1^g) \\
A_2(\text{err}, s_2^{\text{err}}, k_2^{\text{err}}, g, s_2^g, k_2^g) &= \sigma^+(\text{err}, s_2^{\text{err}}, k_2^{\text{err}})\sigma^-(g, s_2^g, k_2^g) \\
A_3(\text{err}, s_3^{\text{err}}, k_3^{\text{err}}, g, s_3^g, k_3^g) &= \sigma^-(\text{err}, s_3^{\text{err}}, k_3^{\text{err}})\sigma^-(g, s_3^g, k_3^g) \\
A_4(\text{err}, s_4^{\text{err}}, k_4^{\text{err}}, g, s_4^g, k_4^g) &= \sigma^-(\text{err}, s_4^{\text{err}}, k_4^{\text{err}})\sigma^+(g, s_4^g, k_4^g) \\
A_5(g, s_5^g, k_5^g) &= \sigma^-(g, s_5^g, k_5^g)
\end{aligned} \tag{8}$$

where $s_i^{\text{err}}, k_i^{\text{err}}$ are the slope and the position in the error direction and $s_i^g, k_i^g$ are the slope and the position in the proximity direction of $A_i$.

Ideally, $\phi$ should assign large weights to high proximity samples producing high errors to trigger shrinking ($A_2$), it should assign large weights to low proximity samples producing low errors to trigger growth ($A_4$), and it should assign zero weights in all other cases ($A_1$ and $A_3$):

$$\phi(\mathbf{p}) = A_5(\mathbf{p}_5) \cdot (A_2(\mathbf{p}_2) + A_4(\mathbf{p}_4)) \tag{9}$$

The parameter values that merge or produce gaps between $A_2$ and $A_4$ cause unwanted behavior. This is avoided by grouping like parameters: $s_2^{\text{err}}$ and $s_4^{\text{err}}$ into $s^{\text{err}}$, $k_2^{\text{err}}$ and $k_4^{\text{err}}$ into $k^{\text{err}}$, $s_2^g$, $s_4^g$, and $s_5^g$ into $s^g$, $k_2^g$ and $k_4^g$ into $k^{\text{in}}$, and $k_5^g$ into $k^{\text{out}}$. The parameters are thus:

$$\begin{aligned}
\mathbf{p} &= (\text{err}, s^{\text{err}}, k^{\text{err}}, g, s^g, k^{\text{in}}, k^{\text{out}}) \\
\mathbf{p}_2 &= (\text{err}, s^{\text{err}}, k^{\text{err}}, g, s^g, k^{\text{in}}) \\
\mathbf{p}_4 &= (\text{err}, s^{\text{err}}, k^{\text{err}}, g, s^g, k^{\text{in}}) \\
\mathbf{p}_5 &= (g, s^g, k^{\text{out}})
\end{aligned} \tag{10}$$

### 3.2 Setting Parameter Values

Given the weighting function $\phi(\mathbf{p})$ (Equation 9) and a reasonable choice of the values for parameters $\mathbf{p}$ (Equation 10), a receptive field can shape itself to cover a region up to an error of $k^{\text{err}}$. The following discussion is based on observations and results of trial learning runs.

The $k^{\text{err}}$ parameter determines which samples are included in the distance metric update. Smaller values result in (generally) smaller fields that approximates the underlying region well, while larger values include more points and thus cover more space. $k^{\text{err}}$ requires tuning, as it affects the overall error, and the size of the fields (and thus their number, given the model allocation strategy in LWPR).

The $k^{\text{in}}$ parameter determines the proximity at which the weights change between exclusion and inclusion. Smaller values cause inclusion of the points with a high proximity, shrinking the field too much. Larger values assign large weights to points with a high error that are at a lower proximity, possibly making the field grow incorrectly. A good value for $k^{\text{in}}$ was found to be around 1 standard deviation, in the interval $[0.30, 0.40]$.

The $k^{\text{out}}$ parameter determines the locality of the field with respect to the distance metric update. Smaller values void the growth, making the fields only

shrink, while larger values enable the fields to observe more of the input space. The update performed as desired when the value of $k^{\text{out}}$ was set in the interval $[0.80, 0.95]$.

Finally, the slope parameters $s^{\text{err}}$ and $s^g$ determine the rate of the transition of other parameters. Smaller values cause smoother, slower updates to the distance metric, while larger values lead to quick, abrupt updates. A reasonable setting for $s^{\text{err}}$ is $\geq 100$, since a sharp threshold between the errors is required. $s^g$ should be set in the range $[30, 50]$, as abrupt updates are not desirable.

## 4 Evaluation

We evaluate our distance metric update by comparing it to the default LWPR distance metric update (Equation 2) on learning two two–dimensional functions (shown in Figure 2) defined in the $[-1, 1]^2$ interval:

1. the *Crossed Ridge* function, containing a mixture of linear and non–linear regions

$$y = \max[\exp(-10x_1^2), \exp(-50x_2^2), 1.25\exp(-5(x_1^2 + x_2^2))], \text{ and}$$

2. the *Sine* function, which is linear in the $(1, -1)$ direction but highly non–linear in the perpendicular $(1, 1)$ direction

$$y = \sin(2\pi(x_1 + x_2))$$

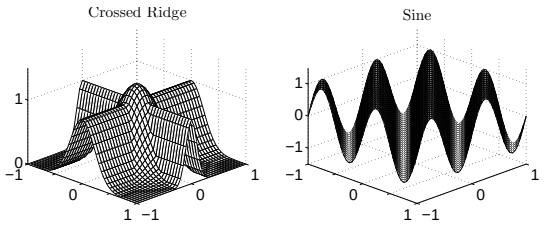



**Fig. 2. The two target functions.**

We do not include the default LLM distance metric update (Equation 3) in the evaluation, as it only aims to provide the coverage of the input space without structuring it.

Two implementations of our distance metric update (Equation 4) are incorporated in the comparison:

1. *CovLWPR*, replacing the default LWPR distance metric update by our covariance update, and
2. *CovLLM*, replacing the default LWPR distance metric update by our covariance update and the default LWPR PLS regression with the LLM regression.

Of interest is the comparison of accuracy, the populations, and the input space structuring between LWPR, CovLWPR, and CovLLM. The metrics include the mean absolute error (MAE), the population size, the visually–judged structuring of the input space, and the averege field volume.

Ten independent runs of the three systems on each function were performed. Each run consisted of 50,000 learning iterations of noiseless samples from a uniform–random distribution in the input space. The values for the non–default parameters used are summarized in Figure 3. The values of $k^{\text{err}}$ used in CovLWPR and CovLLM were 0.0080 for the Crossed Ridge and 0.0350 for the Sine. The performance after learning is shown in Figure 4, and the resulting population plots for the three functions are given in Figure 5.

| | Value | Meaning |
|---|---|---|
| `init_D` | 500 | Initial size of RFs |
| `alpha` | 200 | Distance metric learning rate |
| `w_gen` | 0.05 | RF insertion treshold |
| `w_prune` | 0.75 | RF removal treshold |

**(a) LWPR**

| | Value |
|---|---|
| $k^{\text{in}}$ | 0.4 |
| $k^{\text{out}}$ | 0.8 |
| $s^{\text{err}}$ | 500 |
| $s^{g}$ | 30 |
| $\gamma$ | $5 \times 10^{-4}$ |

**(b) The weighting function $\phi$**

| | Value | Meaning |
|---|---|---|
| $\epsilon_A$ | 0.45 | Linear model learning rate |
| $\epsilon_{out}$ | 0.45 | Output weights learning rate |

**(c) LLM**

**Fig. 3. Values for the non–default parameters used in the evaluation.**

CovLLM and LWPR reach the same error on both functions. CovLLM produces a smaller final population with a greater average field volume than LWPR. This is due to the growth resulting from our distance metric update and the consequent pruning of overlapping fields. The population plots illustrate that the final structure of the CovLLM populations resemble the learned functions, while it is difficult to grasp the structure of the functions from LWPR's populations. LWPR tends to produce circular receptive fields by shrinking them from all directions. Similar characteristics have been reported in Stalph et al. [2010].

CovLWPR is the least accurate of the three systems. Unlike the regression in LLM, the PLS regression retains the history of the training samples seen previously, making it difficult to recover from incorrect shape updates while the regression is still being trained. This restrains the system from learning responsively, and is illustrated both by the higher error and the population plots. Although a rough structure can be identified, the resulting fields are not as slender in the non–linear directions, they are not as long in the linear directions,

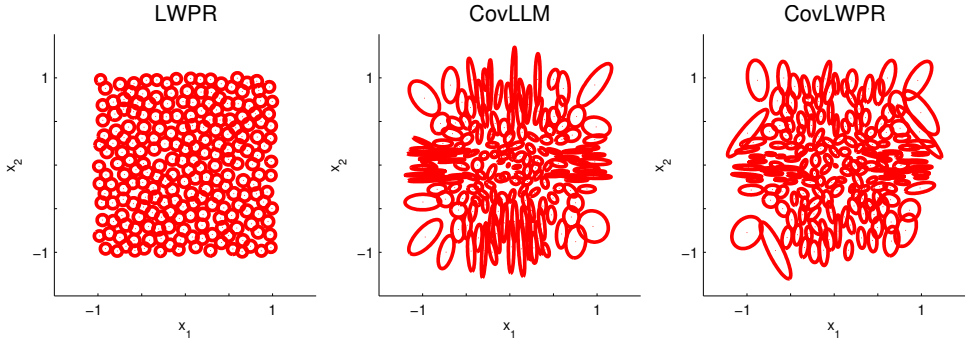
| | MAE | Number of RFs | Average Volume |
|---|---|---|---|
| LWPR | $1.17 \times 10^{-2} \pm 1.34 \times 10^{-7}$ | $220.4 \pm 15.1$ | $1.36 \times 10^{-2} \pm 2.03 \times 10^{-9}$ |
| CovLLM | $1.13 \times 10^{-2} \pm 2.07 \times 10^{-7}$ | $159.8 \pm 20.6$ | $1.58 \times 10^{-2} \pm 3.28 \times 10^{-7}$ |
| CovLWPR | $1.23 \times 10^{-2} \pm 2.65 \times 10^{-7}$ | $172.6 \pm 11.6$ | $1.49 \times 10^{-2} \pm 1.46 \times 10^{-7}$ |

**(a) The Crossed Ridge results**

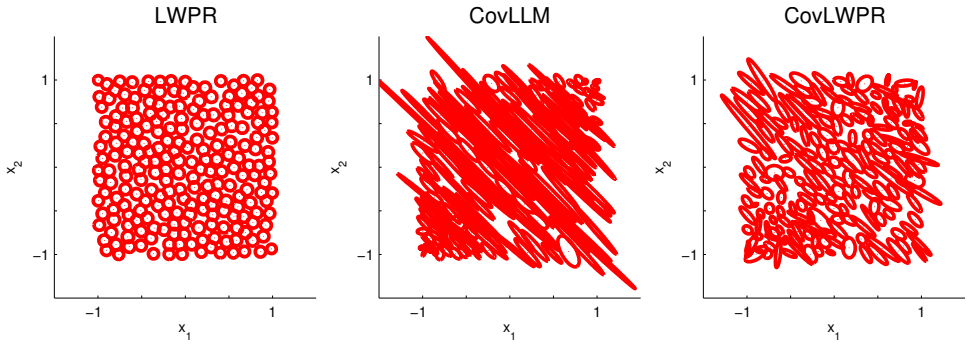
| | MAE | Number of RFs | Average Volume |
|---|---|---|---|
| LWPR | $2.64 \times 10^{-2} \pm 3.80 \times 10^{-7}$ | $220.4 \pm 10.7$ | $1.26 \times 10^{-2} \pm 1.53 \times 10^{-9}$ |
| CovLLM | $2.67 \times 10^{-2} \pm 3.95 \times 10^{-6}$ | $138.3 \pm 25.6$ | $1.91 \times 10^{-2} \pm 4.43 \times 10^{-7}$ |
| CovLWPR | $4.25 \times 10^{-2} \pm 2.07 \times 10^{-5}$ | $187.4 \pm 28.3$ | $1.25 \times 10^{-2} \pm 1.94 \times 10^{-7}$ |

**(b) The Sine results**

**Fig. 4. The prediction errors, population sizes, and field volumes ($E \pm \sigma$) after learning.**




**(a) Populations after learning the Crossed Ridge**




**(b) Populations after learning the Sine**

**Fig. 5. The population plots for a sample run.**

and some fields are misaligned. The amount of history kept can be reduced but, unfortunately, the PLS regression in such cases becomes unreliable, causing the mutual dependency between the shape and the regression to interfere with the ability to learn.

## 5 Conclusion

We introduced a weighted covariance distance metric update for statistics–based LWR algorithms. The initial tests show that the method allows the receptive fields to exploit the local linear substructures of target functions more readily than the default LWPR update. This property may lead to accurate learning with smaller populations, particularly of functions that exhibit large linear substructures. In addition, the structure of the target function can be inferred from model populations.

The future work will focus on studying the relationship between the parameters of the weighting function used in the covariance update and the global error, and the convergence behavior of the distance metric update. We will also evaluate the method in learning higher–dimensional problems.

# Bibliography

C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11:11–73, 1997.

T. Finch. Incremental calculation of weighted mean and variance. *University of Cambridge*, 2009.

D. Ormoneit and T. Hastie. Optimal kernel shapes for local linear regression. In *Advances in Neural Information Processing Systems*, pages 540–546. MIT Press, 1999.

H. Ritter, T. Martinetz, and K. Schulten. *Neural computation and self-organizing maps - an introduction*. Computation and neural systems series. Addison-Wesley, 1992. ISBN 978-0-201-55443-4.

S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10:2047–2084, 1997.

O. Sigaud, C. Salaun, and V. Padois. On-line regression algorithms for learning mechanical models of robots: a survey. *Robotics and Autonomous Systems*, 59 (12):1115–1129, December 2011.

P. O. Stalph, J. Rubinsztajn, O. Sigaud, and M. V. Butz. A comparative study: function approximation with LWPR and XCSF. In *GECCO (Companion)*, pages 1863–1870, 2010.

S. Vijayakumar, A. D'Souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17:2602–2634, 2005.

# Learning in Networks of Similarity Processing Neurons

Lluís A. Belanche

Computer Science School - Dept. of Software
Technical University of Catalonia
Jordi Girona, 1-3 08034, Barcelona, Catalonia, SPAIN
belanche@lsi.upc.edu

**Abstract.** Similarity functions are a very flexible container under which to express knowledge about a problem as well as to capture the meaningful relations in input space. In this paper we describe ongoing research using similarity functions to find more convenient representations for a problem –a crucial factor for successful learning– such that subsequent processing can be delivered to linear or non-linear modeling methods. The idea is tested in a set of challenging problems, characterized by a mixture of data types and different amounts of missing values. We report a series of experiments testing the idea against two more traditional approaches, one ignoring the knowledge about the dataset and another using this knowledge to pre-process it. The preliminary results demonstrate competitive or better generalization performance than that found in the literature. In addition, there is a considerable enhancement in the interpretability of the obtained models.

**Key words:** Similarity representations; Classification; Neural Networks

## 1 Introduction

The intuitive notion of *similarity* is very useful to group objects under specific criteria and has been used with great success in several fields like Case Based Reasoning [1] or Information Retrieval [2]. Interest around purely similarity-based techniques has never faded way; on the contrary, it has grown considerably since the appearance of kernel-based methods [3]. In learning systems, a non-written principle states that similar inputs should have similar outputs for the model to be successful. While this is no guarantee of good performance –specially near class boundaries, where the principle is violated– it certainly is a *sine qua non* condition. If the principle is not true, generalization becomes almost impossible. For a learning system, the trick is then to capture (that is, to *learn*) meaningful similarity relations in relation to the prescribed target variable.

Specific similarity functions from the point of view of data analysis have been used with success since the early days of pattern recognition. Modern modelling problems are difficult for a number of reasons, including dealing with mixtures of data types and a significant amount of missing information [4]. For example,

in the well-known UCI repository [5] over half of the problems contain explicitly declared nominal variables, let alone other data types (*e.g.*, ordinal), usually unreported. In many cases this *heterogeneous* information has to be encoded in the form of real-valued quantities, although there is often enough domain knowledge to characterize the nature of the variables.

The aim of this paper is to demonstrate the learning abilities of simple layered architectures, where the first layer computes a user-defined *similarity function* between inputs and weights. The basic idea is that a combination of partial similarity functions, comparing variables independently, is more capable at capturing the specific properties of an heterogeneous dataset than other methods, which require *a priori* data transformations. An appealing advantage is found in the enhanced interpretability of the models, so often neglected in the neural network community. In order to develop the idea, we propose to compute the similarities among the elements in the learning dataset, and then use a *reduction* method to select a small subset thereof. These selected observations are the *centers* of the first hidden layer. In other words, the first hidden layer is a change of the representation space from the original feature space to a similarity space [6].

## 2 Methodology

### 2.1 Preliminaries

We depart from a training data matrix $D_{N\times d}$ composed of $N$ observations $\mathbf{x}$ described by $d$ variables, plus a target matrix $D_{N\times t}$ containing the known targets of the $N$ observations. For simplicity, in this paper we concentrate in classification problems only (two-class or multiclass) and set $t = 1$.

Given a similarity function $s$, we first compute the associated symmetric similarity matrix $S_{N\times N}$, where $S_{ij} = s(\mathbf{x}_i, \mathbf{x}_j)$. An algorithm is then needed to select a number $d'$ of prototypes, that best represent the learning data in the following sense:

1. the prototypes must be known elements of the input space (observations);
2. all observations that are not prototypes must show a high similarity to (only) one of the prototypes in relation to their similarity to the other prototypes;
3. $d'$ should be set much smaller than $N$.

This is an ideal task for a clustering algorithm, although not all clustering methods are adequate, and certainly alternative techniques could be possible. As an example, artificial immune systems have been used for prototype selection tasks using nearest-neighbor classifiers [7].

The result of this process is a layer of $d'$ units, which we call S-neurons. Any learning method can now operate in the representation space spanned by the layer of $d'$ S-neurons. However, it pays to start using linear methods, both for computational (they are fast) and analytical (they have a single optimum) reasons. It also turns out that the resulting model can be much more interpretable.

### 2.2 Detailed description

Let us represent the observations as belonging to a space $X \neq \emptyset$ as a vector $\mathbf{x}$ of $d$ components, where each component $x_k$ represents the value of a particular feature $k$. A *similarity measure* is a unique number expressing how "like" two observations are, given these features. It can be defined as an upper bounded, exhaustive and total function $s : X \times X \to I_s \subset \mathbb{R}$ such that $I_s$ has at least two different elements (therefore $I_s$ is upper bounded and $s_{max} \equiv \sup_{\mathbb{R}} I_s$ exists).

A basic but very useful S-neuron can be devised using a Gower-like similarity index, well-known in the literature on multivariate data analysis [8]. For any two vector objects $\mathbf{x}_i, \mathbf{x}_j$ to be compared on the basis of feature $k$, a score $s_{ijk}$ is defined, described below. First set $\delta_{ijk} = 0$ when the comparison of $\mathbf{x}_i, \mathbf{x}_j$ cannot be performed on the basis of feature $k$ for some reason; for example, by the presence of missing values, by the feature semantics, etc; $\delta_{ijk} = 1$ when such comparison is meaningful. If $\delta_{ijk} = 0$ for all the features, then $s(\mathbf{x}_i, \mathbf{x}_j)$ is undefined. The partial scores $s_{ijk}$ are defined as follows:

*Binary* (dichotomous) variables indicate the presence/absence of a trait, marked by the symbols + and −. Their similarities are computed according to Table 1, leading to a partial coefficient introduced by Jaccard and well known in numerical taxonomy as the Jaccard Coefficient [9].

Table 1: Similarities for dichotomous (binary) variables.

| | Values of feature $k$ | | | |
|---|---|---|---|---|
| Object $\mathbf{x}_i$ | + | + | − | − |
| Object $\mathbf{x}_j$ | + | − | + | − |
| $s_{ijk}$ | 1 | 0 | 0 | 0 |
| $\delta_{ijk}$ | 1 | 1 | 1 | 0 |

*Categorical* variables can take a number of discrete values, which are commonly known as *modalities*. For these variables no order relation can be assumed. Their *overlap* similarity is $s_{ijk} = 1$ if $x_{ik} = x_{jk}$ and $s_{ijk} = 0$ if $x_{ik} \neq x_{jk}$.

*Real-valued* variables are compared with the standard metric in $\mathbb{R}$: $s_{ijk} = 1 - |x_{ik} - x_{jk}|/R_k$, where $R_k$ is the *range* of feature $k$ (the difference between the maximum and minimum values). The overall coefficient of similarity is defined as the average score over all partial comparisons:

$$S_{ij} = s(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{k=1}^{n} s_{ijk}\delta_{ijk}}{\sum_{k=1}^{n} \delta_{ijk}}$$

Ignorance of the absent elements and normalization by the number of the present ones has been found superior to other treatments in standard data anal-

ysis experiments [10][1]. This coefficient has been extended to deal with other data types, like ordinal and circular variables [11]. Notice that we now have $s_{max} = 1$.

As for the clustering, we choose the PAM algorithm, which partitions data into $k$ clusters, very much like $k$-means. However, PAM offers two advantages: first, the cluster centers (called *medoids*) are chosen *among* the data points; second, the algorithm first looks for a good initial set of medoids and then finds a suboptimal solution such that there is no single switch of an observation with a medoid that will decrease the reconstruction error (the sum of distances of the observations to their closest medoid). The algorithm is fully described in [12].

## 3 Experiments

In this section we report on experimental work in which the previous ideas are applied both to linear learners –logistic and multinomial regression (LogReg and Multinom)– and linear discriminant analysis (LDA) and also to a non-linear learner (a standard SVM using the RBF kernel). All methods are deterministic and use the same data partitions. The smoothing parameter in the RBF kernel is estimated using the *sigest* method, based upon the 10% and 90% quantiles of the sample distribution of $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ [13]; the cost parameter $C$ is set to 1.

All datasets are split into learning and test parts (respecting original partitions, if available). For missing value imputation, we use the Multivariate Imputation by Chained Equations (MICE) method [14], which generates multiple imputations for incomplete multivariate data by Gibbs sampling. This method is attractive because, if the data contains categorical variables, these are also used in the regressions on the other variables.

We study three approaches:

**raw** There is no effort in identifying variable types (all information is considered numerical, and scaled); missing values are either not identified or left as they come (for example, treated as zeros).

**std** All variable types are properly identified; non-numerical information is binarized with a standard dummy code [15]. Missing values are identified and imputed with MICE.

**sim** Same as before with a first layer of S-neurons, as described in Section (2.2); then PAM selects $d' = \lfloor 0.05 \cdot N \rfloor$ prototypes in the learning part. Notice that, in this case, the model has the architecture of a neural network.

### 3.1 Datasets

Some challenging problems have been selected as characteristic of modern modeling datasets because of the diversity in data heterogeneity and the presence of

[1] It is not difficult to realize that this is equivalent to the replacement of the missing similarities by the average of the non-missing ones. Therefore, the conjecture is that the missing values, if known, would not change the overall similarity significantly.

missing values. The problem descriptions and the datasets are taken from the UCI repository [5]. The available documentation has been analyzed for an assessment on the more appropriate treatment. Missing information is also properly identified – see Table 2. The Horse Colic dataset has been investigated with two different targets (variables #23 and #24, resp.).

Table 2: Basic characteristics of the datasets: #Obs (learning, test). Def. (default accuracy), Missing (percentage of missing values). In→Out (no. of inputs and outputs). The last column shows variable types: (R)eal, (N)ominal, or(D)inal.

| Name | #Obs | Def. | Missing | In→Out | Data |
|---|---|---|---|---|---|
| *Pima Diabetes* | 768 (500,268) | 65.1% | 10.6% | 8 → 2 | 8R, 0N, 0D |
| *Horse Colic-23* | 363 (295,68) | 61.4% | 25.6% | 22 → 3 | 7R, 7N,8D |
| *Horse Colic-24* | 364 (296,68) | 63.5% | 25.6% | 22 → 2 | 7R, 7N,8D |
| *Audiology* | 226 (200,26) | 66.3% | 2.1% | 31 → 4 | 0R, 24N, 7D |

**Pima Diabetes**. This is a much studied dataset, in which a population of Pima Indian women living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. In this dataset, most of the variables show impossible zero values (e.g, the diastolic blood pressure), which are actually missing values [15]. Upon careful analysis, it turns out that only 392 out of the 768 observations are unaffected by missing values.

**Horse Colic**. This dataset makes an excellent case study, because of the diversity in data heterogeneity and a significant amount of missing values; it has been used as a paradigmatic example in some textbooks [16]. Each observation is the clinical record of a horse and the variables are specially well documented[2].

**Audiology**. This problem is interesting for many reasons: it is multiclass, has a low number of observations and all variables are categorical (with different numbers of modalities, and some of them ordered)[3]. We have reduced the original 24 classes to 4 by grouping and eliminated non-informative variables.

### 3.2 Results

The results are displayed in Tables 3, 4, and 5. At a first look, it is surprising how the learning methods are able to grasp the task using the **raw** method. In this sense, the **std** method is markedly better for LogReg and the SVM, but not for Multinom or LDA. However, the difference for Multinom is very small; for LDA, the **std** method increases input dimension quite a lot (specially if there are many categorical variables or these have many modalities). The explosion in

[2] This dataset is made available thanks to M. McLeish and M. Cecile (Computer Science Dept., Univ. of Guelph, Ontario, Canada).

[3] Original owner: Professor Jergen at Baylor College of Medicine.

the number of binary variables (due to the dummy coding) changes the data distribution to something extremely non-gaussian, and this causes trouble to LDA (this is specially acute in Audiology). The **sim** method presents similar results for LogReg and the SVM, and much better for both LDA and Multinom.

We would like to point out the good results delivered by linear models, like LogReg –when applicable– and LDA, specially for the **sim** method. On the other hand, few efforts have been devoted to a fine tuning of the SVM models beyond educated guesses, but this issue affects all approaches. Finally, no effort has been put in selecting the optimal number of centers for the **sim** approach.

Table 3: Generalization errors for the **raw** method.

| | LogReg | Multinom | SVM | LDA |
|---|---|---|---|---|
| Pima | 0.201 | 0.187 | 0.194 | 0.187 |
| HorseColic-23 | – | 0.309 | 0.279 | 0.279 |
| HorseColic-24 | 0.176 | 0.162 | 0.162 | 0.162 |
| Audiology | – | 0.231 | 0.154 | 0.269 |
| AVERAGE | 0.189 | 0.222 | 0.197 | 0.224 |

Table 4: Generalization errors for the **std** method.

| | LogReg | Multinom | SVM | LDA |
|---|---|---|---|---|
| Pima | 0.190 | 0.198 | 0.205 | 0.201 |
| HorseColic-23 | – | 0.265 | 0.279 | 0.353 |
| HorseColic-24 | 0.147 | 0.191 | 0.147 | 0.147 |
| Audiology | – | 0.269 | 0.038 | 0.731 |
| AVERAGE | 0.169 | 0.231 | 0.168 | 0.358 |

Table 5: Generalization errors for the **sim** method.

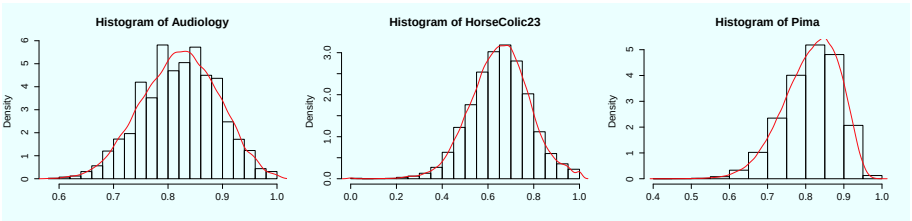
| | LogReg | Multinom | SVM | LDA |
|---|---|---|---|---|
| Pima | 0.183 | 0.190 | 0.194 | 0.175 |
| HorseColic-23 | – | 0.324 | 0.294 | 0.309 |
| HorseColic-24 | 0.162 | 0.176 | 0.176 | 0.191 |
| Audiology | – | 0.115 | 0.000 | 0.038 |
| AVERAGE | 0.172 | 0.201 | 0.166 | 0.178 |

### 3.3 Discussion

When comparing to related previous work, the obtained results are very competitive in relation to those typically reported for these problems, sometimes achieved using very sophisticated techniques. For example, for the Pima dataset, the best reported results are around 20%, topping at 19.8% [15], while typical results are in the range 21%-25% [17]. Our best result is 17.5% using LDA and the **sim** method. Among other causes, this is due to the bad identification or treatment of missing values, something that has been advocated elsewhere [15]. What is more, many (if not most) of these reported results are cross-validation ones, which means that there is no independent assessment of true generalization ability. In our case, the Pima results are reported in a test set that is large, compared to the learning set size (35%). For HorseColic-23, the best reported result seems to be 13.6% [17], while typical results are in the range 14%-23% [18]. Our best result is 14.7%, achieved using three different learners and the **std** method. There seems to be no comparable previous work for HorseColic-24. Finally, for Audiology it is difficult to compare because in this paper we have cleaned the dataset prior to learning; in any event, both the SVM and LDA achieve very good results with the **sim** method.

Another important issue is the distribution of the similarities across the observations and the classes. Fig. 1 shows this distribution for the different datasets. It can be seen that in all cases similarities are rather high and well-behaved (fairly symmetrical, unimodal). Given the assumed relation between similarity computations and learning ability, we computed the intra-class similarities. These were: Cochlear (0.847), Mixed (0.822), Normal (0.914) and Other (0.794) for Audiology, No (0.811) and Yes (0.788) for Pima, Died (0.646), Euthanized (0.634) and Lived (0.673) for HorseColic-23 and No (0.688) and Yes (0.673) for HorseColic-24. These numbers not only reflect how relatively compact the different classes are, they also indicate the hardness for a learning method based on distances or similarities (however they are computed). For example, HorseColic-23 and HorseColic-24 show markedly less compact classes. The relation to overall performance and to class-by-class performance is left for a further dedicated study.

Fig. 1: Similarity distributions for the different datasets.



## 4 Conclusions and Future Work

A shortcoming of many existent learning methods –and, in particular, neural networks– is the difficulty of adding prior knowledge to the model in a principled way. Current practice assumes that input vectors may be faithfully represented as a point in $\mathbb{R}^d$, and the geometry of this space is meant to capture the meaningful relations in input space. There is no particular reason why this should be the case, at least not with small numbers of hidden neurons. This paper has described ongoing research on more flexible learning frameworks, offering means for the injection of prior knowledge, and permitting a natural extension to operate in problems with non-numerical data types and showing missing values.

When talking about real problems, however, accuracy may not tell the whole picture about a model. Other performance criteria include development cost, interpretability and usability.

- The *cost* here refers to how much pre-processing effort we need in order to build the model. Undoubtedly, all but the **raw** method require more analysis time compared to doing (almost) nothing. Prior to learning the variable types must be identified and coded properly; for the S-neurons, suitable similarity measures must be chosen, using available background knowledge;
- The *interpretability* refers to the complexity of the obtained model in human terms. It is generally believed that accuracy and interpretability are in conflict [19]. In the present case, the methods based on similarity have a clear advantage in this case when combined with a linear learner: the prediction is a weigthed combination of the similarity of the input to a selected (and small) subset of prototypes. This framework resembles that of an SVM; however, in SVMs the kernel is providing an implicit transformation of the input space rather than a purely similarity-based representation. Moreover, the chosen similarity should be a valid kernel function;
- Finally, models must be *useful* in practice: in a real deployment of the model, new and unseen observations emerge which we need to classify, which display the same variable types and may contain missing values (that certainly could not be imputed at learning time). The similarity approaches are able to face this situation without further effort.

Current lines of research include the extension to new data types (a suitable similarity measure is needed in each case) and the design of formal measures to compute the relation between overall and, particularly, intra-class similarities with class distribution itself. A measure of similarity that is maximized for observations of the same class and minimized for observations of different classes is envisaged via the introduction of weights into the comparisons. This approach would permit the optimization of the similarity measure to some extent. Another important issue is the selection of the best centers, which could be performed in a supervised way. For example, performing a separate clustering per class and merging the results, or by using GLVQ methods [20]. This latter family of algorithms makes good use about the classes to which the input vector and the

winning codevector belong at prototype selection time. It is conjectured that a supervised reduction method will deliver better modelling results when coupled with subsequent stages of the method.

**Acknowledgments.** Partially supported by the MICINN project BASMATI (TIN2011-27479-C04-03) and by SGR2009-1428 (LARCA).

## References

1. Osborne, H., Bridge, D. Models of similarity for case-based reasoning. Interdisciplinary Workshop on Similarity and Categorisation, pp. 173–179 (1997)
2. Baeza-Yates, R., Ribeiro, B. *Modern information Retrieval*. ACM Press, New York (1999)
3. Shawe-Taylor, J., Cristianini, N. *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press (2004)
4. Little, R., Rubin, D. *Statistical Analysis with Missing Data*. Wiley, (2009)
5. Bache, K. and Lichman, M. UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`. Irvine, CA: University of California (2013)
6. Pekalska, E. *The Dissimilarity representations in pattern recognition. Concepts, theory and applications.* ASCI Dissertation Series no. 109. Delft University of Technology, Delft, (2005)
7. Garain, U. Prototype reduction using an artificial immune model. *Pattern Analysis and Applications* 11:3-4, 353-363 (2008)
8. Gower, J.C. A General Coefficient of Similarity and Some of Its Properties. *Biometrika*, 27(4), pp. 857–871 (1971)
9. Sokal, R. R. and Michener, C. D *Principles of Numerical Taxonomy*. San Francisco: W.H. Freeman (1963)
10. Dixon, J.K. Pattern recognition with partly missing data. *IEEE Trans. on Systems, Man and Cybernetics*, 9: 617-621, (1979)
11. Pavoine, S., Vallet, J., Dufour, A.B., Gachet, S., Daniel, H. On the challenge of treating various types of variables: application for improving the measurement of functional diversity. *Oikos*, 118(3), pp. 391–402, (2009)
12. Kaufman, L. and Rousseeuw, P.J. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, (1990)
13. Caputo, B., Sim, K., Furesjo, F., Smola, A. Appearance-based object recognition using SVMs: which kernel should I use? NIPS Workshop on Statistical methods for computational experiments in visual processing and computer vision, (2002)
14. van Buuren, S., Groothuis-Oudshoorn, K. mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, 45(3):1–67, (2011)
15. Ripley, B. *Pattern Recognition and Neural Networks*. Cambridge Univ. Press, (1996)
16. Xu, R., Wunsch, D. *Clustering*. Wiley-IEEE Press, (2008)
17. Torre, F. Boosting Correct Least General Generalizations. Technical Report GRAppA-0104, (2004)
18. Yang, J., Parekh, R., Honavar, V. DistAI: An Inter-pattern Distance-based Constructive Learning Algorithm. *Intelligent Data Analysis* 3, pp. 55-73 (1999)
19. Breiman, L. Statistical Modeling: The Two Cultures. *Statistical Science* 16 (3), 199-231, (2001)
20. Pal, N.R., Bezdek, J.C., Tsao, E. Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Trans. Neural Networks*, 4(4) pp.549-557, (1993)

# Human Activity Classification with Online Growing Neural Gas

Maximilian Panzner, Oliver Beyer, and Philipp Cimiano

Semantic Computing Group, CITEC, Bielefeld University,
obeyer@cit-ec.uni-bielefeld.de
http://www.sc.cit-ec.uni-bielefeld.de

**Abstract.** In this paper we present an online approach to human activity classification based on Online Growing Neural Gas (OGNG). In contrast to state-of-the-art approaches that perform training in an offline fashion, our approach is online in the sense that it circumvents the need to store any training examples, processing the data on the fly and in one pass. The approach is thus particularly suitable in life-long learning settings where never-ending streams of data arise. We propose an architecture that consists of two layers, allowing the storage of human actions in a more memory efficient structure. While the first layer (feature map) dynamically clusters Space-Time Interest Points (STIP) and serves as basis for the creation of histogram-based signatures of human actions, the second layer (class map) builds a classification model that relies on these human action signatures. We present experimental results on the KTH activity dataset showing that our approach has comparable performance to a Support Vector Machine (SVM) while performing online and avoiding to store examples explicitly.

**Keywords:** artificial neural networks, online growing neural gas, human action recognition, space-time, online classification

## 1 Introduction

The recognition and classification of human activity is important in many application domains including smart homes [1], surveillance systems [2], ambient intelligence [3], etc. In particular, we address the task of classifying human activity into a given set of activity types on the basis of video data, sequences of 2D images in particular. State-of-the-art approaches extract features from space-time volumes, e.g. *space-time interest points (STIP)* as introduced by Ivan Laptev [4]. Their advantage lies in their compact and robust representation of human actions, as they reduce the input space by identifying local feature points. Training a human action classifier model based on STIPs typically includes the clustering (with e.g. k-Means) of video features to yield bag-of-visual-words clusters that can be used to derive a histogram-based representation of a video clip by indicating the number of STIPs being assigned to each cluster. Then a classifier (e.g. SVM) is trained to learn to classify image sequences into a set of given human activity types.

There are several drawbacks in this classical approach. First of all, the approach requires an architecture that comprises heterogeneous algorithms, e.g. a feature extractor, a clustering algorithm and a classification algorithm. An architecture that is more uniform and compact relying on one algorithm would be simpler, easier to implement and thus preferable. Further, the approach is not online, requiring to store a number of examples in memory or on disk in order to recompute the cluster and retrain the classifier at regular intervals.

To circumvent these limitations, we present a new architecture which is based on Online Growing Neural Gas (OGNG), which has been presented earlier [5]. In this paper we present a two-layer architecture which consists of two maps that we call *feature map* and *class map*, respectively. In the first layer (feature map), STIPs are dynamically clustered according to their similarity in the feature space, yielding a growing set of "visual words" that can also change over time. A *human activity signature (HAS)* for each space-time volume is then formed by a histogram indicating the activity of each prototype / neuron in the feature map. In the second layer (class map), space-time volumes are clustered by their HAS and labelled according to the corresponding activity.

Our contributions can be listed as follows:

- **Online classification**: We provide an architecture that grows incrementally and is capable of processing space-time volumes in an online fashion as new data arrives.
- **Compact model**: We provide a compact human activity model as both layers are based on STIPs and OGNG which represent the high dimensional input space in a low dimensional map.
- **Uniform architecture**: We provide an architecture which is uniformly based on OGNG, in contrast to existing approaches that rely on more heterogeneous structures.

We compare our architecture to the classical architecture proposed by Laptev [4] based on a k-means based feature discretization as well as an SVM-based classification, showing that our approach yields comparable results to the latter approach, while proposing a uniform architecture based on two OGNG maps and circumventing the need to store examples to process them offline. Our approach is thus suitable in a life-long learning setting, in which there is a never-ending data stream that needs to be processed on-the-fly as in the applications mentioned above.

The paper is structured as follows: in Section 2 we describe the OGNG algorithm in order to make this paper self-contained. In Section 3 we describe our online approach to human action classification with OGNG, including a description of the features we use. In Section 4 our experiments, including the methodology of our evaluation, the used baseline and our results are described. We then conclude and provide an overview over the related work in Section 5.

## 2 Online Growing Neural Gas (OGNG)

Online Growing Neural Gas (OGNG) as introduced by Beyer and Cimiano [5], extends Growing Neural Gas to an online classifier by integrating additional online labeling and prediction strategies. In the following we will briefly describe the OGNG algorithm. The algorithm is depicted in Algorithm 1 and modifications are highlighted. A detailed description of OGNG and a comparison of several online labeling and prediction strategies can be found in Beyer and Cimiano. [5].

**Algorithm 1** Online Growing Neural Gas (OGNG)

1: Start with two units $i$ and $j$ at random positions in the input space.
2: Present an input vector $x \in R^n$ from the input set or according to input distribution.
3: Find the nearest unit $n_1$ and the second nearest unit $n_2$.
4: Assign the label of $x$ to $n_1$ according to the present labeling strategy.
5: Increment the age of all edges emanating from $n_1$.
6: Update the local error variable by adding the squared distance between $w_{n_1}$ and $x$.

$$\Delta error(n_1) = |w_{n_1} - x|^2$$

7: Move $n_1$ and all its topological neighbours (i.e. all the nodes connected to $n_1$ by an edge) towards $x$ by fractions of $e_b$ and $e_n$ of the distance:

$$\Delta w_{n_1} = e_b(x - w_{n_1})$$

$$\Delta w_n = e_n(x - w_n)$$

for all direct neighbours of $n_1$.
8: If $n_1$ and $n_2$ are connected by an edge, set the age of the edge to 0 (refresh). If there is no such edge, create one.
9: Remove edges with their age larger than $a_{max}$. If this results in nodes having no emanating edges, remove them as well.
10: If the number of input vectors presented or generated so far is an integer or multiple of a parameter $\lambda$, insert a new node $n_r$ as follows:
Determine unit $n_q$ with the largest error.
Among the neighbours of $n_q$, find node $n_f$ with the largest error.
Insert $a$ new node $n_r$ halfway between $n_q$ and $n_f$ as follows:

$$w_r = \frac{w_q + w_f}{2}$$

Create edges between $n_r$ and $n_q$, and $n_r$ and $n_f$. Remove the edge between $n_q$ and $n_f$.
Decrease the error variable of $n_q$ and $n_f$ by multiplying them with a constant $\alpha$. Set the error $n_r$ with the new error variable of $n_q$.
11: Decrease all error variables of all nodes $i$ by a factor $\beta$.
12: If the stopping criterion is not met, go back to step (2).

In steps 1-3, the network is initialized and the first winner $n_1$ and second winner $n_2$, according to a presented stimulus, are determined. In step 4 we assign the label of our stimulus $\xi$ to the winner neuron $n_1$ according to the selected labeling strategy. The selected labeling strategy is the *relabeling method (relabel)*, because of its simplicity and effectiveness as shown in Beyer and Cimiano [5]. In steps 5-7, the age of all edges emanating from $n_1$ are incremented by one. Furthermore, the local error gets updated, and $n_1$ and its topological neighbors $n$ are adapted towards the stimulus by the learning rates $e_b$ (for $n_1$) and $e_n$ (for the topological neighbors). In steps 8-9, $n_1$ and $n_2$ get connected by an edge and

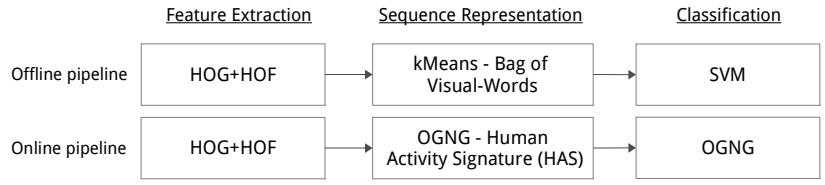



**Fig. 1.** Online and offline processing pipelines.

the edges with an age larger than $a_{max}$ are removed. In step 10, a new neuron $n_r$ is introduced between the neuron $n_q$ with the largest local error and its neighbor $n_f$, having the largest error of its neighborhood. We only insert a new neuron if $n_{max}$, the maximal number of neurons per class, is not exceeded for the class of $\xi$. In step 11 all error variables are decreased by the factor $\beta$. In the last step 12, the algorithm continues with step 2 if the stopping criterion (mostly when a predefined maximum number of neurons has been reached) is met.[1]

## 3 Human Activity Classification with OGNG

In this section we describe our two-layer online approach to human activity classification that exploits topological maps - Growing Neural Gas maps in particular - at both layers. We call our approach *Online Human Action Classifier* (OHAC). A typical processing pipeline in human activity recognition is comprised of the following three steps:

1. **Extraction of video features:** Extraction of distinctive features from a sequence of video frames.
2. **Representation of video features:** Calculating video signatures from the extracted features that capture the similarity between different video sequences.
3. **Classification of video signatures:** Training a classifier to learn to recognize a set of previously observed classes of human actions.

Our algorithm covers steps 2 and 3 of the typical processing pipeline. For the first step, the extraction of video features, we use HOG+HOF features calculated around sparsely detected spatio-temporal interest points (STIPS) as proposed by Laptev [4]. The detection of the points and the extraction of the feature descriptors are completely online in the sense that no global information is required. STIPs are detected as local maxima of a Harris-Corner-Function extended into the spatio-temporal domain. The spacial Harris-Corner-Function characterizes the "'cornerness"' of an image point by the strength of its intensity gradients in all directions. The spatio-temporal Harris-Corner-Function responds to points in space-time where the motion of local image structures is non-constant. Aside from sensor noise and other disruptions, the motion of local image structures is

[1] For our experiments we additionally introduce new neurons for novel categories in the learning process. Furthermore, we only stop inserting new neurons when the maximum number of neurons is reached, instead of stopping the training process.

primarily the result of forces acting on the corresponding physical objects. Hence the local neighbourhood of the detected points can be expected to provide meaningful information about motion primitives in that point of space-time. The combination of STIPs and HOG+HOF feature descriptors has already shown promising results in several synthetic [4] and real world datasets [4, 6].

### 3.1 Static Human Action Classifier according to Laptev

As baseline we use an offline approach proposed by Laptev [4]. This approach uses k-means for clustering and an SVM for classification. The video sequence is represented as a *bag of visual words* histogram. The visual vocabulary is built by clustering the feature vectors in the training set into a predetermined number of clusters.

**Algorithm 2** Static Human Action Classifier according to Laptev

1: Cluster the training set into a predetermined number of clusters using kMeans.
2: Initialize Histogram $H$ with one entry for each prototype vector.
3: Present an input vector $x \in S$ from the extracted features of the video sequence.
4: Find the nearest prototype $p_x$ to the presented input.
5: Increment the corresponding histogram entry $p_x$ by one.
6: repeat step 3 until all features are processed.
7: Normalize $H$ using $L_1$ norm
8: Train the SVM with the normalized histogram and the label $l$ of the current sequence.

In step 1 the *visual vocabulary* is built in advance by clustering all feature vectors in the training set into a predetermined number of clusters[2] using kMeans. Each cluster stands with its prototype vector for one distinct *visual word* in the *visual vocabulary*. Steps 3-6 iterate over the feature vectors in the current training sequence. Each feature vector is assigned to its nearest prototype vector and incorporated into the histogram. The resulting histogram represents the given video sequence as a bag of visual words. To compensate for different counts of features in different sequences, the histogram is normalized using the $L_1$ norm. In step 8 the SVM is trained with the normalized histogram and the corresponding label of the current sequence.

### 3.2 Online Human Action Classifier (OHAC)

The Online Human Action Classifier consists of two independent OGNG networks. The first network is what we call the *feature map*. It utilizes the OGNG algorithm for clustering incoming data in feature space. The second network is called the *class map*, as it uses the label information from a training sequence to assign class labels to the nodes in the network according to the *relabel* strategy presented in section 2. Video sequences are represented as *human activity signatures* (*HAS*), which are represented by a histogram indicating the activity of each neuron in the *feature map*, while iterating through the respective video sequence.

[2] We use $k = 4000$ for the number of clusters following Laptev [6]

**Algorithm 3** Online Human Action Classifier (OHAC)

1: Initialize the *feature-* and *class-maps*.
2: Initialize the $HAS$ histogram $H$ with two (number of initial nodes in the *feature map*) entries $h_1 = 0, h_2 = 0$.
3: Present an input vector $x \in S$ from the extracted features of the video sequence.
4: Find the nearest unit $n_x$ from the *feature map*.
5: **if** node $n_x$ is new **then**
6: insert new entry into the histogram at position $x$
7: **end if**
8: Increment the corresponding histogram entry $h_x$ by one.
9: repeat step 3 until all features are processed.
10: Normalize $H$ using $L_2$ norm.
11: Update the OGNG *class map* with the normalized $HAS$ histogram $H$ and label $l$ of the video sequence.

Algorithm 3 is initialized by first initializing the OGNG network and creating an empty *bag of visual words* histogram (steps 1-2). The histogram starts with two entries, one for each of the two initial nodes in the OGNG network. In steps 3-5 the next input vector $x$ is presented to the *feature map*, and the node $n_x$ that is closest to the presented stimulus is located. If the located node $n_x$ is newly inserted into the *feature map*, a new histogram entry is created at position $x$. In step 8 the histogram entry at position $x$ is incremented. When all input vectors in the sequence are processed, the histogram is normalized by $L_2$ norm. The normalization compensates for different numbers of extracted feature vectors in different video sequences. The normalized histogram is then used to update the *class map* OGNG network with the label $l$ of the video sequence. To predict the label of a previously unseen video sequence $S$, all input vectors $x \in S$ are incorporated into the histogram $H$ by the number of the *feature map* node $n_x$ nearest to them (steps 3-5). The histogram is then normalized and the label $l$ is predicted by the *class map* according to the *single linkage strategy* (see Beyer and Cimiano [5]).

## 4 Experiments and Evaluation

### 4.1 Dataset

As a dataset for evaluation we use the KTH human action dataset [7] [3]. This video database consists of six categories of human actions (walking, jogging, running, boxing, hand waving and hand clapping). All actions are performed several times by 25 subjects in four different scenarios (outdoors, outdoors with scale variations, outdoors with different clothes and indoors). Each combination of 25 subjects, 6 actions in 4 scenarios gives a total of 600 video files.

### 4.2 Evaluation Methodology

We evaluated the accuracy of OHAC and our baseline on the KTH human action dataset. We generated 15 training and test sets by separating the 600 video clips

[3] Examples of the six actions are shown on the following website http://www.nada.kth.se/cvap/actions/

into 300 training examples and 300 test examples for each set. We furthermore took care that each of the six categories was equally distributed in the training and test set. We averaged our accuracy results over the 15 runs and also determined a best and worst result of the 15 runs.

The OGNG parameters are set as follows: insertion parameter $\lambda = 50$; maximum age $a_{max} = 120$; adaptation parameter for winner $e_b = 0.3$; adaptation parameter for neighbourhood $e_n = 0.0018$; error variable decrease $\alpha = 0.5$; error variable decrease $\beta = 0.0005$. We also allowed a maximum of 4000 neurons for the feature map and 200 for the class map.

### 4.3 Results

Our results are depicted in Table 1. The matrices show the confusion matrix of our Baseline (left) and OHAC (right). Thereby, each row represents the to be classified human action category, while each column holds the percentage of examples that have been classified into the category written on top of the matrices. Overall, OHAC achieves an averaged accuracy of 93%, while our Baseline holds an accuracy of 95%. We performed a t-test and could not prove that the results of both approaches are statistically significant. We thus consider the classification performance of the algorithms to be comparable. The confusion matrix shows that both algorithms are having issues to distinguish between jogging and running, which is intuitively understandable as we as humans also would consider those two activities to be closer to each other compared to the other four. Furthermore, it is interesting that OHAC slightly less confuses the human action categories of "hand clapping" and "running" with 93.2% and 72.2% compared to 89.8% and 66.8% of our Baseline. It also should be mentioned that the confusion of OHAC is spread more uniformly compared to the Baseline approach, which could be explained by the generative approach of OHAC compared to the discriminative character of our Baseline and i.e. of the underlying SVM classifier.

## 5 Related Work & Conclusion

In this paper we have presented a novel human activity classifier model based on Online Growing Neural Gas (OGNG). The model provides a compact architecture and consists of two layers, allowing the storage of human actions in a more memory efficient structure. While the first layer (feature map) dynamically clusters STIPs and serves as base for the creation of histogram-based signatures of a human action, the second layer (class map) builds a classification model that builds upon those human action signatures. The advantage of this novel architecture lies in its ability to perform a human action classification task online as the model stepwise adapts to new data and grows incrementally. The uniform character of the algorithm is desirable, as that its simplicity allows an easy implementation and integration into existing systems. In most cases, heterogeneous offline human action recognition approaches have been proposed [4,

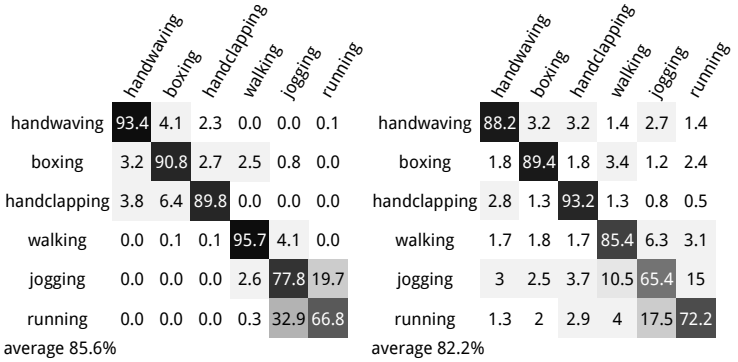



**Table 1.** Confusion matrix of our Baseline (left) and OHAC (right) on the KTH human action database, averaged over 15 runs.

8], that generate action signatures by clustering STIPs with static clustering algorithms (such as k-Means) and classifying them with an offline classifier that needs to be retrained as new data arrives. We have experimentally shown on the KTH dataset that our approach reaches comparable performance to a classical offline SVM-based classification approach while performing online and avoiding the need to store training examples explicitly, thus being suitable in lifelong stream data settings.

## References

1. Marie Chan, Eric Campo, Daniel Estève, and Jean-Yves Fourniols. Smart homes-current features and future perspectives. *Maturitas*, 64(2):90–97, 2009.
2. M Valera and SA Velastin. Intelligent distributed surveillance systems: a review. In *Proceedings of the International Conference on Computer Vision, Image and Signal Processing*, volume 152, pages 192–204. IET, 2005.
3. Eric J Pauwels, Albert Ali Salah, and Romain Tavenard. Sensor networks for ambient intelligence. In *Proceedings of the IEEE Workshop on Multimedia Signal Processing (MMSP)*, pages 13–16. IEEE, 2007.
4. Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.
5. Oliver Beyer and Philipp Cimiano. Online labelling strategies for growing neural gas. In *Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)*, pages 76–83. Springer, 2011.
6. Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.
7. Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Proc. of the Int. Conference on Pattern Recognition (ICPR)*, volume 3, pages 32–36. IEEE, 2004.
8. Alper Yilmaz and Mubarak Shah. Actions sketch: A novel action representation. In *IEEE Computer Society Conference in Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 984–989. IEEE, 2005.

# About the Equivalence of Robust Soft Learning Vector Quantization and Soft Nearest Prototype Classification

David Nebel and Thomas Villmann

Computational Intelligence Group,
Department for Mathematics/Natural & Computer Sciences,
University of Applied Sciences Mittweida, Mittweida, Germany
email: {nebel,villmann}@hs-mittweida.de

**Abstract.** In this paper we consider the Robust Soft Learning Vector Quantization and the Soft Nearest Prototype Classification as variants of the learning vector quantization paradigm proposed by T. Kohonen for prototype based classification models. We show that, although separately introduced, the latter one is a special case of the other model.

## 1 Introduction

Robust Soft Learning Vector Quantization (RSLVQ) and Soft Nearest Prototype Classification (SNPC) are two famous learning vector quantizers for prototype based classification learning [6,5]. Both approaches can be understood as probabilistic variants of the Bayesian motivated but heuristic learning vector quantization schemes introduced by T. Kohonen [1]. Whereas RSLVQ maximizes a log-likelihood cost function, SNPC uses a soft variant of the classification error to quantify the classification performance. Thus, these algorithms can be seen as alternatives to the generalized learning vector quantization approach (GLVQ), which takes an approximation of the classification error as cost function for a LVQ-like prototype based classification system [4].

Although both methods, RSLVQ and SNPC, were introduced independently, there are inherent similarities in methodology: RSLVQ as well as SNPC can be seen as probabilistic models of classification based on prototypes. These prototypes are interpreted as centers of local probability models of the class distribution.

In this paper we show that SNPC is mathematically equivalent to RSLVQ for a particular parameter setting in RSLVQ. Thus, both models can be identified as the same mathematical model. For this purpose, a unifying description is presented. In consequence, SNPC can be optimized via an expectation maximization scheme as recently proposed for RSLVQ [2] and GLVQ [3].

## 2 The RSLVQ model

In following we briefly describe the RSLVQ as introduced by SEO&OBERMAYER in [5]. We suppose data points $x_i \in \mathbb{X}$ $(i = 1, ..., N)$ and prototypes $\theta_j \in \Theta$ $(j = 1, ..., M)$. Further, let $c(\cdot)$ be the formal class labeling function, which assigns to each data point the class label $c(x_i)$. Analogously, $c(w_j)$ returns the class label of the prototype.

In RSLVQ, the data density is modeled by a Gaussian mixture approach: Let $p(\theta_j) = \frac{1}{M}$ be the prior probability that data points are generated by the $j$-th single component of the mixture. Let

$$p(x_i|\theta_j) = \frac{1}{\sqrt{(2\pi\sigma^2)^D}} \exp\left(-\frac{(x_i - \theta_j)^2}{2\sigma^2}\right) \tag{1}$$

be the conditional probability that the $j$-th component of the Gaussian mixture generates the data point $x_i \in \mathbb{R}^D$. Then the probability density for the data point $x_i$ is given by

$$p(x_i|\Theta) = \sum_{j=1}^{M} p(\theta_j)p(x_i|\theta_j)$$

and the conditional probability

$$p_c(x_i|\Theta) = \sum_{\{j:c(x_i)=c(\theta_j)\}} p(\theta_j)p(x_i|\theta_j)$$

is the probability density that a data point $x_i$ is generated by the mixture model for the correct class. The cost function of the RSLVQ is based on the likelihood ratio

$$L_r = \prod_{i=1}^{N} \frac{p_c(x_i|\Theta)}{p(x_i|\Theta)}. \tag{2}$$

The resulting cost function to be maximized in RSLVQ is obtained as

$$K_{RSLVQ}(\mathbb{X}, \Theta) = \sum_{i=1}^{N} \log\left(\frac{p_c(x_i|\Theta)}{p(x_i|\Theta)}\right) \tag{3}$$

which can be optimized using a (stochastic) gradient ascent scheme, see [5]. Recently, a generalized expectation maximization (gEM) scheme for RSLVQ was proposed as an alternative to the gradient learning [2], if only relational data or

dissimilarities between data are available. For this purpose, the cost function (3) can be rewritten as

$$K_{RSLVQ}(\mathbb{X}, \Theta) = \sum_{i=1}^{N} \log \left( \sum_{\{j:c(x_i)=c(\theta_j)\}} g(x_i, \theta_j) \right) \tag{4}$$

with the functions

$$g(x_i, \theta_j) = \frac{\exp\left(-\frac{(x_i-\theta_j)^2}{2\sigma^2}\right)}{\sum_{k=1}^{M} \exp\left(-\frac{(x_i-\theta_k)^2}{2\sigma^2}\right)} \,. \tag{5}$$

In the following we use this formulation of the cost function to show its relation to the SNPC.

## 3 SNPC revisited: equivalence to RSLVQ

The SNPC classifier is a soft variant of nearest prototype classification (NPC) approach. For the NPC the cost function

$$C_{\text{err}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} \left[1 - \delta\left(c(\theta_j) = c(x_i)\right)\right] \delta(\theta_j = \theta_{bmu(i)}) \tag{6}$$

counts the misclassifications whereby

$$\delta(x = y) = \begin{cases} 1, & x = y \\ 0, & else \end{cases}$$

is the Kronecker symbol and

$$bmu(i) = \underset{l=\{1,\ldots,M\}}{\arg\min}\ d\left(x_i, \theta_l\right)$$

is the index of the best matching unit (prototype) for a given data point $x_i$ with respect to a predefined dissimilarity measure $d\left(x_i, \theta_l\right)$, frequently chosen as the squared Euclidean distance. The original cost function of SNPC is the probabilistic version

$$K_{SNPC}(\mathbb{X}, \Theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} \left[1 - \delta\left(c(\theta_j) = c(x_i)\right)\right] P\left(\theta_j | x_i\right) \rightarrow min$$

of (6) with the conditional probability

$$P\left(\theta_j|x_i\right) = \frac{\exp\left(-d(x_i,\theta_j)\right)}{\sum_{k=1}^{M}\exp\left(-d(x_i,\theta_k)\right)}$$

that the data point $x_i$ is assigned to the prototype $\theta_j$ [5]. We remark at this point, that these probabilities are structurally equivalent to the functions $g\left(x_i,\theta_j\right)$ from (5) Equivalently to minimization of $K_{SNPC}(\mathbb{X},\Theta)$, we could maximize

$$\begin{aligned} K(\mathbb{X},\Theta) &= \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\delta\left(c(\theta_j)=c(x_i)\right)P\left(\theta_j|x_i\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{\{j:c(\theta_j)=c(x_i)\}} P\left(\theta_j|x_i\right) \end{aligned}$$

instead of $K_{SNPC}(\mathbb{X},\Theta)$. Further, the maximization does not change under an application of a monotonically increasing function. Hence, we consider the equivalent cost function

$$\hat{K}_{SNPC}(\mathbb{X},\Theta) = \frac{1}{N}\sum_{i=1}^{N}\log\left(\sum_{\{j:c(\theta_j)=c(x_i)\}} P\left(\theta_j|x_i\right)\right) \tag{7}$$

Yet, this is exactly the cost function of RSLVQ (4) if we set $\sigma = \sqrt{\frac{1}{2}}$ in (4) and ignore the scaling factor $\frac{1}{N}$ in (7). Hence, SNPC can be seen as a special case of RSLVQ.

## 4 Conclusion

In the present paper we have shown the mathematical equivalence between separately introduced RSLVQ and SNPC, which were both invented by Seo&Obermayer. Using the recently in [2] published result for RSLVQ, it follows immediately, that SNPC also can be maximized using the gEM optimization procedure presented in [2] for RSLVQ.

## References

1. Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 1995. (Second Extended Edition 1997).
2. D. Nebel, A. Gisbrecht, B. Hammer, and T. Villmann. Learning supervised generative models for dissimilarity data. In *Proceedings of Conference on Neural Information Processing Systems (NIPS)*, page submitted. MIT press, 2013.
3. D. Nebel and T. Villmann. A median variant of generalized learning vector quantization. In *Proceedings of International Conference on Neural Information Processing (ICONIP)*, page submitted. Springer, 2013.
4. A. Sato and K. Yamada. Generalized learning vector quantization. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8. Proceedings of the 1995 Conference*, pages 423–9. MIT Press, Cambridge, MA, USA, 1996.
5. S. Seo, M. Bode, and K. Obermayer. Soft nearest prototype classification. *IEEE Transaction on Neural Networks*, 14:390–398, 2003.
6. S. Seo and K. Obermayer. Soft learning vector quantization. *Neural Computation*, 15:1589–1604, 2003.

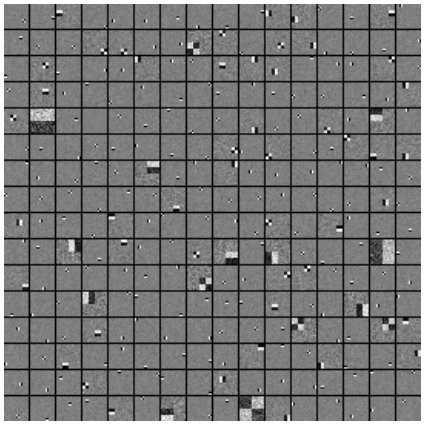
# Learning Orthogonal Bases for k-Sparse Representations

Henry Schütze, Erhardt Barth, Thomas Martinetz

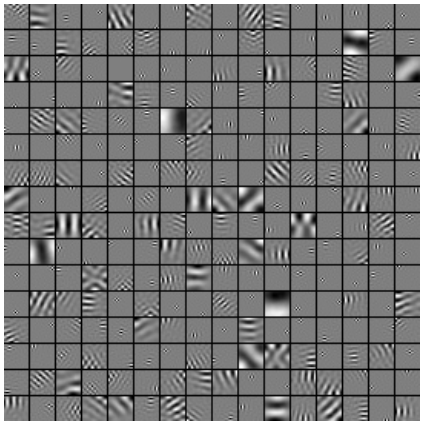
Institute for Neuro- and Bioinformatics, University of Lübeck
Ratzeburger Allee 160, 23562 Lübeck, Germany
schuetze@inb.uni-luebeck.de

Sparse Coding aims at finding a dictionary for a given data set, such that each sample can be represented by a linear combination of only few dictionary atoms. Generally, sparse coding dictionaries are overcomplete and not orthogonal. Thus, the processing substep to determine the optimal $k$-sparse representation of a given sample by the current dictionary is *NP*-hard. Usually, the solution is approximated by a greedy algorithm or by $l_1$ convex relaxation. With an orthogonal dictionary, however, an optimal $k$-sparse representation can not only be efficiently, but exactly computed, because a corresponding $k$-sparse coefficient vector is given by the $k$ largest absolute projections.

In this paper, we present the novel online learning algorithm Orthogonal Sparse Coding (OSC), that is designed to find an orthogonal basis $U = (\boldsymbol{u}_1, ..., \boldsymbol{u}_d)$ for a given data set $X \in \mathbb{R}^{d \times L}$, such that for any $k \in \{1, ..., d\}$, the optimal $k$-sparse coefficient vectors $A \in \mathbb{R}^{d \times L}$ minimize the average representation error $E = \frac{1}{dL} \|X - UA\|_{\mathrm{F}}^2$. At each learning step $t$, OSC randomly selects a sample $\boldsymbol{x}$ from $X$ and determines an index sequence $h_1, ..., h_d$ of decreasing overlaps $|\boldsymbol{u}_{h_i}^{\mathrm{T}} \boldsymbol{x}|$ between $\boldsymbol{x}$ and the basis vectors in $U$. In the order of that sequence, each



(a) Learned basis from 1,000 *synthetic* image patches of size 16×16 pixel.



(b) Learned basis from 20,000 *natural* image patches of size 16×16 pixel.

Fig. 1: Basis patches learned with OSC.

basis vector $\boldsymbol{u}_{h_i}$ is updated by the Hebbian learning rule $\Delta\boldsymbol{u}_{h_i} = \varepsilon_t(\boldsymbol{u}_{h_i}^{\mathrm{T}}\boldsymbol{x})\boldsymbol{x}$ with a subsequent unit length normalization. After each basis vector update, $\boldsymbol{x}$ and the next basis vector $\boldsymbol{u}_{h_{i+1}}$ to be adapted are projected onto the orthogonal complement $\mathrm{span}(\{\boldsymbol{u}_{h_1}, ..., \boldsymbol{u}_{h_i}\})^{\perp}$ wherein the next update takes place.

We applied OSC to $(i)$ 1,000 synthetic ($k$=50)-sparse patches of size 16×16 pixel, randomly generated with a 2D Haar basis, and $(ii)$ 20,000 natural image patches of size 16×16 pixel, that were randomly sampled from the first image set of the nature scene collection [1] (308 images of nature scenes containing no man-made objects or people). The basis patches learned by OSC are shown in Figure 1 and demonstrate that OSC reliably recovers the generating basis from synthetic data (see Figure 1a). Figure 1b illustrates that the OSC basis learned on the natural image patches resembles a wavelet decomposition, and is distinct from PCA, DCT, and Haar bases.

In Figure 2, the average $k$-term approximation performance of the OSC basis is compared with PCA, DCT, Haar and JPEG 2000 wavelets on the natural image patch data set. For this data set, OSC yields a consistently better k-term approximation performance than any of the alternative methods.

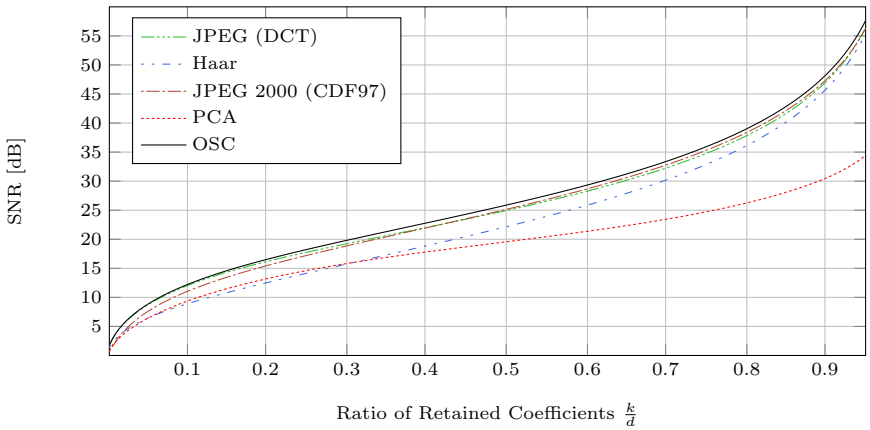



Fig. 2: Average $k$-term approximation performance of 20,000 natural image patches of size 16×16 pixel.

**Acknowledgement.**

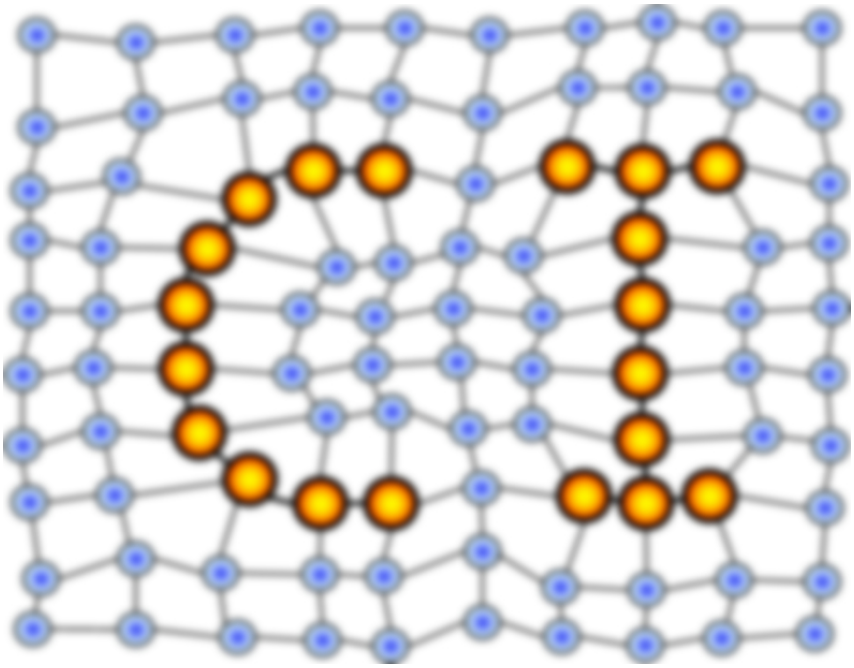
The research is funded by the DFG Priority Programme SPP 1527, grant number MA 2401/2-1.

## References

1. Wilson S. Geisler and Jeffrey S. Perry. Statistics for optimal point prediction in natural images. *Journal of Vision*, 11(12), October 2011.

# MACHINE LEARNING REPORTS

Report 02/2013




## Impressum

Machine Learning Reports ISSN: 1865-3960

▽ Publisher/Editors

Prof. Dr. rer. nat. Thomas Villmann
University of Applied Sciences Mittweida
Technikumplatz 17, 09648 Mittweida, Germany
• http://www.mni.hs-mittweida.de/

Dr. rer. nat. Frank-Michael Schleif
University of Bielefeld
Universitätsstrasse 21-23, 33615 Bielefeld, Germany
• http://www.cit-ec.de/tcs/about





▽ Acknowledgments

We would like to thank the reviewers for their time and patience.